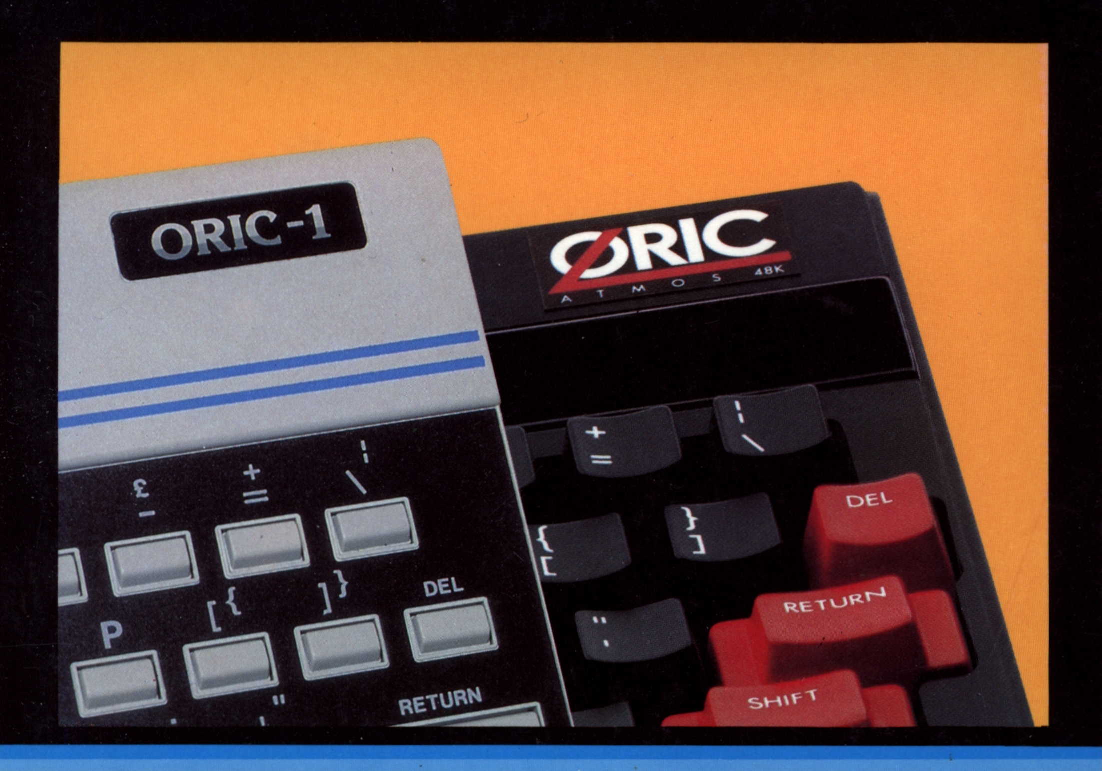
P. GUEULLE

PILOTEZ VOTRE ORIC

1 + ATMOS



MICRO SYSTEMES

ETSF

Collection
POCHE-INFORMATIQUE
dirigée par ALAIN TAILLIAR
directeur de la rédaction de MICRO-SYSTEMES

PILOTEZ VOTRE ORIC

COLLECTION E.T.S.F. MICRO-SYSTEMES

1 - A. VILLARD et M. MIAUX, *Un microprocesseur pas à pas*
2 - A. VILLARD ET M. MIAUX, *Systèmes à microprocesseur*
3 - P. GUEULLE, *Maîtrisez votre ZX 81*
4 - E. FLOEGEL, *Du Basic au Pascal*
5 - P. COURBIER, *Vous avez dit Basic ?*
6 - M. MARCHAND, *Vous avez dit micro ?*
7 - P. GUEULLE, *Pilotez votre ZX 81*
8 - M. JACQUELIN, *La micro-informatique et son A-B-C*
9 - M. OURY, *Maîtrisez le TO 7*

COLLECTION POCHE-INFORMATIQUE

1 - G. ISABEL, *50 programmes pour ZX 81*
2 - P. GUEULLE, *Montages périphériques pour ZX 81*
3 - C. GALAIS, *Passeport pour Applesoft*
4 - R. BUSCH, *Passeport pour Basic*
5 - M. ROUSSELET, *Mathématiques sur ZX 81*
6 - C. GALAIS, *Passeport pour ZX 81*
7 - G. PROBST, *50 programmes pour CASIO FX-702 P et FX-801 P*
8 - G. PROBST, *60 programmes pour CASIO PB-100*

ISBN 2-85535-067-0 ISSN 0759-9498

PATRICK GUEULLE
Ingénieur EFREI

PILOTEZ VOTRE ORIC

Diffusion :
EDITIONS TECHNIQUES ET SCIENTIFIQUES FRANÇAISES
2 à 12, rue de Bellevue, 75940 Paris Cedex 19

SOMMAIRE

Chapitre 6 : ***L'ORIC imprimeur***

Chapitre 7 : ***L'ORIC robot ou à la découverte des interfaces***

Chapitre 8 : ***Quelques idées concrètes***

Chapitre 9 : ***Pour aller plus loin***

Avant-propos

Les machines ORIC appartiennent résolument à ce qu'il est convenu d'appeler la « seconde génération » d'ordinateurs familiaux. Par rapport à ce pionnier que fut le ZX 81, elles offrent une image couleur, la haute résolution graphique, un jeu de caractères beaucoup plus complet (et redéfinissable), un véritable synthétiseur de sons et un clavier mécanique.

Le tout pour un prix qui laisse rêveur, même si l'on choisit la version 48 K, de très loin la plus répandue.

Depuis son lancement, l'ORIC 1 a connu un succès considérable malgré un certain nombre d'anomalies de son BASIC. C'est dire à quel point les qualités de cet ordinateur ont été appréciées par ses utilisateurs, dont l'accueil s'est finalement révélé positif.

« On ne change pas une équipe qui gagne » disent les sportifs, et le nouvel ORIC ATMOS illustre bien ce vieil adage. Sous un aspect extérieur totalement différent, on retrouve en fin de compte exactement la même machine, dont les faiblesses ont été pour la plupart éliminées.

Cet ouvrage s'adressant à la fois aux possesseurs d'ORIC 1 et aux heureux détenteurs d'ORIC ATMOS, nous ne chercherons pas à cacher les bizarreries de la première machine, dont de très nombreux exemplaires resteront encore longtemps en circulation. Les propriétaires d'ATMOS n'en seront que plus heureux de constater que leur ordinateur trace, lui, des cercles bien ronds !

Quant aux adeptes de l'ORIC 1, qu'ils se souviennent « qu'un homme averti en vaut deux » et que « faute avouée est à moitié pardonnée » : des artifices de programmation souvent fort simples permettent de tout faire rentrer dans l'ordre. Les « grands débutants » de l'informatique trouveront ainsi dans ce livre une initiation très progressive au langage BASIC, mettant très tôt à contribution les talents artistiques des machines ORIC.

Les habitués d'autres machines découvriront quant à eux les informations leur permettant de se « convertir » à l'ORIC dans les meilleurs délais, à côté de programmes « prêts à l'emploi » pour diverses applications autrement plus « sérieuses » que les jeux « à grand spectacle ».

Bien que très doué pour la « mise en scène », l'ORIC dispose en effet, dans ses deux versions, d'une excellente interface cassette et d'une prise permettant le raccordement des imprimantes les plus variées.

La porte est ainsi grande ouverte à toutes sortes d'applications « bureautiques », tandis que quelques accessoires peu coûteux suffisent pour que l'ORIC ou l'ATMOS prennent en charge la surveillance ou le pilotage des équipements extérieurs les plus divers, depuis le téléphone du foyer jusqu'à un projecteur de diapositives en passant par la chaudière du chauffage central...

Remarque importante

Dans le courant de cet ouvrage, les machines ORIC 1 et ORIC ATMOS seront nommées ORIC dans tous les cas où leur fonctionnement est identique.

L'emploi des dénominations ORIC 1 ou ATMOS signalera donc des comportements différents, le lecteur étant invité à tenir compte des seuls propos se rapportant à l'appareil dont il dispose.

Chapitre 1

Prise de contact avec l'ORIC

L'ORIC 1 et l'ORIC ATMOS se présentent tous deux sous la forme d'un élégant boîtier dont la face avant regroupe cinquante-sept véritables touches mécaniques. Seule la forme de ces touches varie d'une version à l'autre, celle de l'ATMOS rappelant davantage celles d'une machine à écrire alors que l'ORIC 1 évoque plutôt un grand clavier de calculatrice.

Une impression générale de robustesse se dégage de cette machine, et se confirme lorsque l'on examine le jeu de connecteurs dont dispose la face arrière : pas de compromis en matière de qualité comme en témoigne l'usage de connecteurs professionnels à broches dorées. Un gage de fiabilité des extensions que l'on raccordera tôt ou tard à l'ordinateur !

L'intérieur de la machine, domaine interdit aux possesseurs d'appareils sous garantie, confirme ce sentiment de qualité d'exécution : l'ORIC est un ordinateur fait pour durer... .

Mise en service de la machine

Un écran TV est le seul périphérique indispensable à la prise de contact avec l'ORIC. Divers accessoires disponibles en option permettent l'utilisation de récepteurs français ou de moniteurs noir ou couleur.

C'est, bien entendu, sur un récepteur couleur que l'ORIC donnera sa pleine mesure, mais bien des choses sont faisables sur un écran noir et blanc.

La prise d'antenne UHF, située à l'arrière de la machine, ne peut être raccordée qu'à un récepteur multistandard 625 lignes (PAL en couleur), ou à un récepteur noir et blanc français modifié (retourner la diode de détection vidéo).

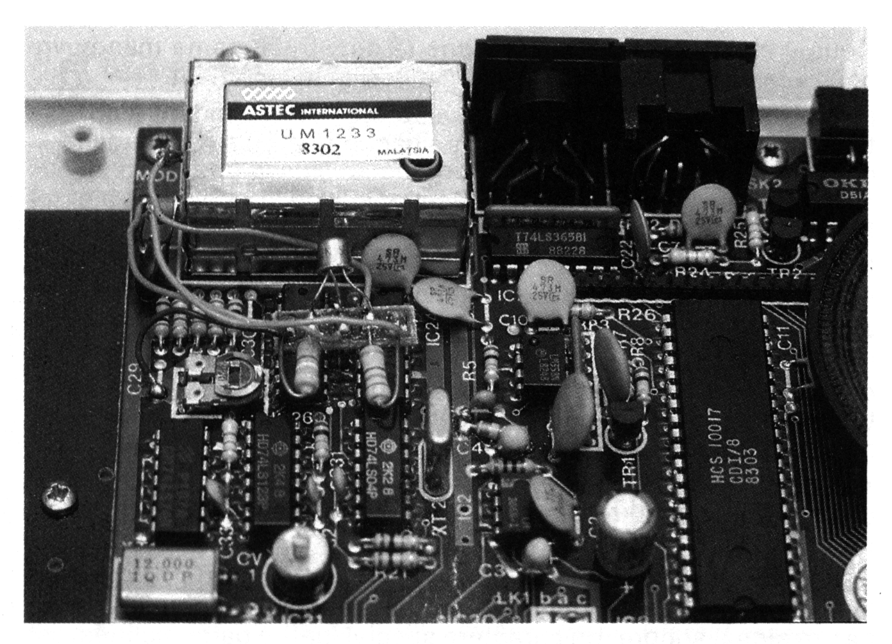
Dans le cas général, on utilisera un adaptateur, le type le plus répandu étant, bien sûr, le cordon *péritélévision.* Cet accessoire, peu coûteux, permet un raccordement direct du socle à cinq broches de l'ORIC à la prise spéciale dont sont munis tous les téléviseurs couleur récemment achetés en France.

Ce branchement direct garantit la meilleure qualité d'image puisque l'ordinateur attaque directement la partie « moniteur » du recepteur, évitant de la sorte tous les circuits de réception. Il est cependant indispensable qu'une tension extérieure soit appliquée au téléviseur afin que celui-ci se trouve commuté en mode « péritélévision ». La plupart des cordons *Péritel* sont livrés munis d'un petit bloc secteur ou d'une pile. A défaut, on reliera le fil blanc sortant de la fiche *Péritel* à la broche n° 33 du grand connecteur de l'ORIC. On veillera, bien sûr, à éviter tout court-cicuit avec une broche voisine.

Un autre type de cordon serait à utiliser pour relier L'ORIC à un simple *moniteur* muni d'une entrée dite « R.V.B. ». Le principal avantage d'un moniteur sur un téléviseur est que, ne recevant pas les programmes TV, cet appareil est exonéré de la redevance télévision !

Ces branchements effectués, on raccordera au secteur le bloc d'alimentation de la machine, puis sa petite fiche dans la douille correspondante de l'ORIC (juste à côté du grand connecteur).

Il est important de noter que l'ordre inverse ne permettrait pas à l'ordinateur de démarrer correctement. On se gardera donc de commander le bloc secteur par un interrupteur 220 V quel qu'il soit. On peut, par contre, munir avantageusement d'une « olive » le fil basse tension allant à la machine.



Le modulateur UHF d'origine ne peut convenir qu'à un récepteur TV acceptant le standard PAL. Le transistor « volant » a été ajouté par l'auteur pour permettre l'attaque directe d'un téléviseur noir et blanc français (possesseurs d'ordinateurs sous garantie s'abstenir !).

Quelques secondes après la mise sous tension de l'ORIC, les formes inquiétantes défilant sur l'écran TV doivent céder la place à un message « de service » se terminant par le mot anglais « Ready » (prêt).

Cette indication reviendra souvent par la suite, indiquant que la machine est « à vos ordres ».

Ne résistons donc pas plus longtemps à la tentation d'utiliser le clavier, et frappons quelques mots.

Chaque appui sur une touche produit un petit « clic » dans le haut-parleur situé sous la machine, ce qui évite à l'opérateur d'avoir constamment à surveiller l'écran. Les grands nerveux pourront faire cesser ce bruitage en appuyant, n'importe quand, sur la touche CTRL (contrôle) puis, sans la lâcher, sur la touche F. En effet, une touche pressée en même temps que CTRL est interprétée comme un *ordre de service* et non comme la frappe d'un caractère.

Nous y reviendrons plus loin dans le détail. La même manœuvre pourra rendre la voix au clavier à n'importe quel moment.

Il est facile de constater que le caractère correspondant à chaque touche s'affiche à l'écran lors de chaque pression. Cependant, si nous maintenons une touche enfoncée plus d'une seconde, le caractère se répétera à une cadence rapide, ce qui s'avèrera bientôt fort utile.

L'ordinateur semble donc se comporter en vulgaire machine à écrire, mais à quelques exceptions près :

• La touche de majuscules (SHIFT) est sans effet pour les lettres. La mention CAPS (capitales) en haut et à droite de l'écran signale cette particularité, dont on peut s'affranchir en pressant CTRL et T. Un jeu complet de minuscules devient disponible tant que SHIFT n'est pas enfoncée.

• La touche DEL (DELete, soit effacement), permet d'annuler le caractère situé immédiatement à gauche du petit carré noir appelé *curseur*.

• Si l'on frappe plus de 79 caractères (espaces compris), un retour à la ligne est automatiquement effectué, annoncé un peu à l'avance par un signal sonore. Un caractère spécial (petite barre inclinée) est ajouté à la ligne tronquée.

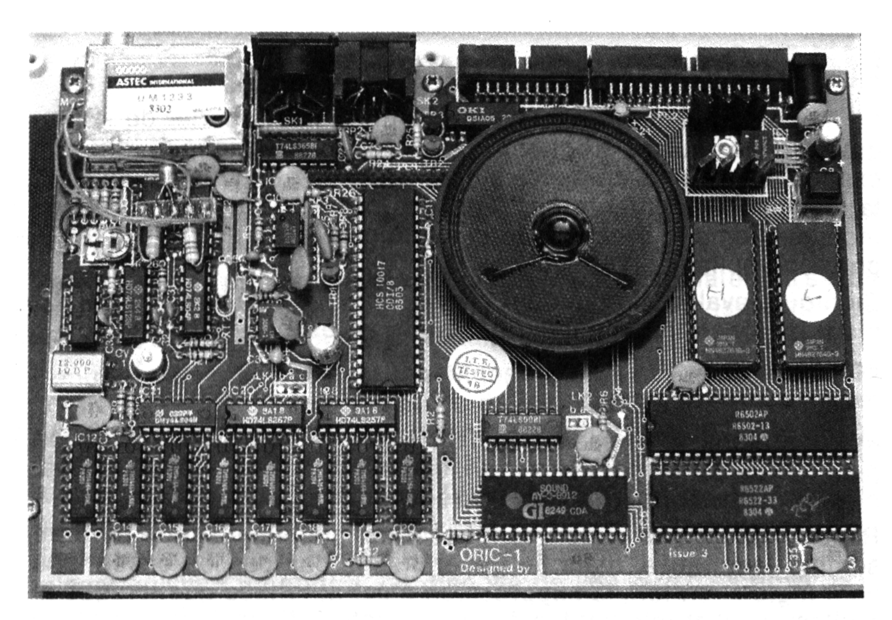
• Enfin et surtout, la touche RETURN (retour chariot) se comporte de façon très spéciale. On pourrait s'attendre à un simple retour à la ligne, comme lorsque l'on ramène à droite le chariot d'une machine à écrire. Essayons !

Si nous frappons, par exemple :

JE M'APPELLE ORIC
puis si nous pressons RETURN, le texte suivant apparaît :

?SYNTAX ERROR
Ready
Cela signifie tout simplement qu'un ordinateur n'est pas une machine à écrire, et que la touche RETURN possède un rôle beaucoup plus important que le simple ordre de retour à la ligne.

Chaque fois que l'on presse RETURN, on *donne l'initiative à la machine,* en lui indiquant que le message que l'on a frappé à son intention est terminé. Il peut s'agir d'un ordre, de renseignements, etc., mais dans tous les cas, l'action sur RETURN informe l'ordinateur que c'est à lui de travailler. Seulement, les ordres doivent être donnés à la machine dans un langage bien précis (le BASIC), et en respectant un certain nombre de règles impératives (la SYNTAXE).



Vue générale des circuits de l'ORIC : du beau travail, laissant espérer une robustesse à toute épreuve...

JE M'APPELLE ORIC n'est pas une instruction BASIC reconnue par la machine, qui informe donc l'opérateur de son refus. Elle manifeste cependant une certaine bonne volonté en ajoutant « Ready », ce qui indique qu'elle ne voit pas d'inconvénient à ce qu'un nouvel ordre, si possible vraisemblable, lui soit soumis.

Essayons donc de frapper :

PRINT "JE M'APPELLE ORIC"

puis pressons RETURN. Le texte situé entre guillemets se trouve imprimé à la suite sur l'écran, suivi de Ready.

Expliquons nous :

Le mot PRINT (frappé lettre à lettre en majuscules) est l'un des *mots-clé* du langage BASIC : il correspond à un ordre que la machine sait exécuter, à savoir l'impression d'un texte sur l'écran. Ce texte doit être libellé entre guillemets à la suite du mot PRINT.

L'action sur RETURN commande à la machine d'exécuter l'ordre immédiatement, puis de se remettre à notre disposition en inscrivant Ready. En effet, nous sommes encore dans le mode de fonctionnement le plus simple de la machine, le *mode commande.*

L'ORIC en mode commande

Le mode commande ne relève pas à proprement parler de la programmation : qui dit programme dit liste d'instructions devant être exécutées les unes après les autres, alors qu'en mode commande, la machine se contente de traiter les ordres qui lui sont donnés individuellement, sans en garder la moindre trace une fois la besogne accomplie.

Bien que ravalant l'ordinateur au modeste rang de calculatrice, ce mode de fonctionnement permet quelques essais intéressants.

Frappons donc :

? "JE M'APPELLE ORIC"

et constatons en pressant RETURN que le point d'interrogation peut remplacer le mot PRINT qui, très souvent employé, peut devenir fastidieux à orthographier trop souvent.

Recommençons en supprimant le dernier guillemet : cette fioriture semble facultative, mais ne généralisons pas car les exceptions sont légion : en fait, la suppression n'est autorisée que lorsque le guillemet est strictement le dernier caractère d'une ligne.

En agissant sur CTRL T, introduisons des minuscules dans le texte, puis, courageusement, dans le mot PRINT : erreur de syntaxe ! nous répond la machine, et, en effet, il est obligatoire d'orthographier tous les mots-clé exclusivement en majuscules. C'est pour cette raison que le clavier est normalement bloqué en capitales, et que le déblocage par CTRL T se fait sous la responsabilité de l'opérateur, lorsqu'il éprouve vraiment le besoin d'incorporer des minuscules à un texte. Il est vivement conseillé de revenir au mode CAPS le plus rapidement possible.

Tentons maintenant de lancer (par RETURN) l'ordre suivant :

PRINT A (ou encore ?A)

La machine n'imprime pas la lettre A, mais 0 (zéro) avant de répondre Ready.

En fait, comme nous n'avons pas placé la lettre A entre guillemets, la machine l'a interprété comme une *variable numérique.*

En mathématiques, il est d'usage de donner des noms à des grandeurs (X, Y, Z, a, b, c, etc.), et les ordinateurs font de même.

Nous demandons donc à l'ORIC d'imprimer la valeur de la variable A, et il répond zéro puisqu'aucune valeur n'a été spécifiée auparavant.

Faisons donc :

A = 5 (puis RETURN)

Et ensuite :

PRINT A (ou ?A) puis RETURN

Cette fois la valeur 5 est imprimée, preuve que la machine l'avait bien conservée en mémoire.

Voici l'une des principales caractéristiques de l'ordinateur, à savoir la possibilité de stocker en mémoire un nombre considérable de variables et de les retrouver à l'instant.

Contrairement à d'autres machines (SINCLAIR par exemple), l'ORIC ne reconnaît que les noms de variables à deux caractères, par exemple A1, A2, GH, etc. Rien n'interdit de définir des noms plus longs, mais tous les caractères suivant le second seront ignorés, et la machine ne fera aucune différence entre *variable* et *vapeur* par exemple.

En pratique, les 26 lettres de l'alphabet suffisent souvent, et en les faisant suivre d'un chiffre, on dispose de 260 variables possibles, ce qui est déjà beaucoup !

Il n'est guère conseillé d'employer des combinaisons de deux lettres, car un certain nombre d'entre elles sont refusées par la machine, notamment lorsqu'elles correspondent à des assemblages existant dans des mots-clé du BASIC. TO et OR ne sont que deux exemples débouchant ainsi sur une erreur de syntaxe.

Si les variables numériques peuvent ainsi porter des noms, les textes jouissent du même privilège, avec les mêmes conditions que les nombres, mais aussi avec les mêmes limites.

Faisons par exemple :

A$ = "ORIC" (et RETURN)

Puis :

PRINT A$ (ou ?A$) et RETURN

C'est le mot ORIC qui s'inscrira à l'écran.

Le signe $ est ainsi caractéristique de tout ce qui concerne le traitement des textes, l'un des domaines de prédilection des ordinateurs.

Notons que la longueur d'un texte ainsi stocké dans une variable nommée CHAÎNE ne peut excéder 255 caractères, ce qui est peu comparé à d'autres machines. Là encore, des artifices permettent

de repousser cette limite à volonté, notamment par mise bout à bout de plusieurs chaînes :

Faisons donc :
A$ = "ORIC" RETURN
B$ = "POUR LA VIE" RETURN

Puis :

PRINT A$ + B$ RETURN

Un léger défaut de présentation apparaît, puisque nulle part n'a été prévu un espace de séparation. On aurait pu ajouter un espace à la fin de A$ ou un au début de B$, mais il est tout aussi simple de faire :

PRINT A$ + " " + B$ RETURN

Le signe + est bien évidemment interprété autrement lorsqu'il s'applique à des variables numériques.

Faisons par exemple :

PRINT 2 + 3 RETURN

Nous obtenons 5 car la machine *effectue* systématiquement toutes les opérations qu'elle rencontre sur son chemin. Toutes, sauf bien sûr celles qui ne sont que présentes dans un texte, comme le démontre l'exemple suivant :

PRINT "2 + 3" + " = 8" RETURN

qui montre bien qu'un ordinateur fait très exactement ce qu'on lui dit de faire, même si la chose est complètement absurde. N'accusons pas trop vite la machine lorsqu'un résultat semble aberrant...

Mystères et merveilles de l'ordre PRINT

Jusqu'à présent, nous avons massivement utilisé l'ordre PRINT pour faire apparaître sur l'écran le résultat de nos manipulations.

En effet, avec LPRINT qui joue le même rôle vis-à-vis d'une éventuelle imprimante, c'est PRINT qui est le plus souvent utilisé lorsqu'il s'agit de faire rendre compte à l'ordinateur.

Il n'a pas manqué d'occasions, jusqu'ici, de constater que l'écran se comporte très exactement comme le rouleau de papier d'un téléscripteur, toute l'image montant d'un cran lorsque l'impression va à la ligne. Egalement, les ordres exécutés précédemment ne dis-

paraissent de l'écran que lorsqu'ils "débordent par le haut", alors que la mémoire centrale de la machine les a oubliés depuis longtemps.

Il est souvent utile de "faire place nette" au moyen d'un ordre CLS (puis RETURN) ou grâce à un CTRL L, ce qui permet de reprendre l'impression tout en haut d'un écran vide.

Un ordre PRINT isolé imprime ce qu'il a à imprimer à gauche de l'écran, puis revient à la ligne, comme l'indique le déplacement du curseur.

On peut, cependant, souhaiter faire apparaître des caractères en un point quelconque de l'écran. La fonction PRINT AT, rencontrée sur d'autres machines, s'appelle ici PLOT.

L'écran comporte 27 lignes de 39 caractères, mais la colonne de l'extrême gauche est interdite à l'impression.

On peut imprimer un caractère quelconque n'importe où sur l'écran par la commande suivante :

PLOT colonne, ligne, "caractère"   RETURN

soit, par exemple :

PLOT 19,13,"+"   RETURN

pour placer un signe plus à peu près au centre de l'écran. On peut même placer un mot, une phrase, un résultat de calcul, mais il faut veiller à ne pas dépasser les bords de l'écran, car de bien étranges phénomènes se produiraient !

Essayez, par exemple :

PLOT C,L,"ORIC"   RETURN

avec des valeurs de C comprises entre 1 et 38, L variant de Ø à 26. C = 36, par exemple, ne manque pas d'intérêt !

Sachez, de toute façon, que quelles que soient les images apparaissant sur l'écran, quelles que soient les manœuvres, même absurdes, exécutées au clavier, vous ne mettrez jamais la machine en danger : au pire vous la bloquerez, et il existe un petit bouton dessous permettant de sortir de cette fâcheuse situation. Dans les cas extrêmes, la coupure de l'alimentation est un remède radical.

L'ORIC en mode PROGRAMME

Même si le mode commande permet déjà des effets intéressants, un ordinateur est avant tout destiné à travailler en mode *programme.*

Un programme est une suite logique d'*instructions* assez voisines des ordres servant en mode commande, mais *numérotées* afin que leur exécution s'opère dans une chronologie parfaitement définie.

Par ailleurs, un programme présent en machine peut être exécuté plusieurs fois de suite, car il ne s'efface pas, une fois son rôle terminé, comme une vulgaire commande directe.

Frappons par exemple :

10 PRINT "ORIC" puis RETURN

et remarquons que rien n'est exécuté. La ligne n° 10 du programme est simplement mise en mémoire. Pour lancer ce (très court !) programme, il est d'usage de frapper RUN, puis RETURN.

Recommençons, et constatons que le programme peut être exécuté aussi souvent que nécessaire. Effaçons l'écran par un CTRL L, puis lançons à nouveau le programe : il est toujours là !

Pour en reprendre connaissance, il faudrait le *lister,* en faisant : LIST RETURN.

Pour s'en débarrasser, il suffirait de faire : NEW RETURN, et même en plus un CLS RETURN pour vider aussi l'écran. Notons bien, cependant, que même si le programme est encore visible sur l'écran après NEW, il ne peut plus être exécuté par RUN, car il a quitté la mémoire centrale. Souvenons-nous que l'écran doit être considéré comme une feuille de papier, et seulement comme tel. Il n'est nullement l'image du contenu de la mémoire de la machine !

Frappons de nouveau notre ligne 10, puis les deux suivantes :

```
20 GOTO 10
5 CLS
```

Lançons un LIST (RETURN) et constatons que les lignes ont été classées par ordre de numéros. Elles seront donc exécutées dans ce même ordre.

Faisons RUN, et constatons que le mot ORIC se répète très vite jusqu'à remplir l'écran, qui venait d'être effacé, du haut jusqu'en bas. En fait, la machine imprime sans arrêt des "ORIC" mais l'écran décale en même temps vers le haut tout son ancien contenu. Tout

au plus un léger scintillement de la ligne du bas trahit-il ce défilement ultra-rapide.

En fait, le programme "boucle", car la ligne 20 déclenche le retour à la ligne 10 qui, dès son exécution, laisse le programme atteindre la ligne 20, et ainsi de suite.

Seule la ligne 5, placée avant la boucle, ne sera exécutée qu'une seule fois.

Il est fréquent que des programmes "bouclent" ainsi, volontairement ou non. Pour les arrêter, il suffit de faire CTRL C ou, dans les cas graves, d'actionner le poussoir "RESET" caché sous la machine. Le programme n'est pas effacé pour autant, comme un LIST permet de s'en convaincre.

Ajoutons la ligne suivante :

```
15 WAIT 50
```

et relançons le programme, sérieusement ralenti, puisque le nombre placé après le mot-clé WAIT ordonne à la machine d'attendre un nombre égal de centièmes de seconde, soit ici, une demi-seconde à chaque exécution de la "boucle". Arrêtons le programme, et modifions ainsi la ligne 10 :

```
10 PRINT "ORIC";
```

Pour ce faire, il existe deux possibilités :

- la refrapper entièrement, puis presser RETURN qui déclenchera l'échange standard ;
- la corriger au moyen de l'éditeur de lignes.

A l'aide des touches fléchées situées de part et d'autre de la barre d'espacement, amener le curseur devant la ligne à corriger. Appuyer alors sur CTRL puis, sans relâcher cette touche, presser A autant de fois qu'il le faudra pour que le curseur parvienne à l'endroit où le point-virgule doit être inséré. Lâcher alors CTRL et A, et presser la touche du point-virgule suivie de RETURN. Un listage montrera que la correction a bien été faite.

Oh, une idée ! transformons par le même moyen "ORIC" en "ORAC". Seulement, il y a là un piège : lorsque le curseur arrive sur le I, il faut bien sûr insérer le A, mais il faut refaire CTRL A *jusqu'à la fin de la ligne*, faute de quoi... mais essayez donc !

En fait, cet éditeur est relativement pratique lorsqu'il s'agit de prolonger une ligne, ou d'en modifier certains caractères. Il s'avère par contre franchement odieux dans les cas où il est nécessaire d'ajouter des caractères dans le courant d'une ligne : mieux vaut alors la frapper à neuf !

Encore une idée ! Supprimons la ligne 15 : il suffit pour cela de faire 15 RETURN, ce qui est fort commode surtout si la ligne est longue.

Lançons notre petit programme, qui doit revêtir, après toutes ces modifications, l'aspect suivant :

```
5 CLS
10 PRINT "ORAC" ;
20 GOTO 10
```

Le point-virgule, sous des dehors assez insignifiants, est capable de bouleverser complètement le fonctionnement de l'ordre PRINT, en annulant le retour à la ligne en fin d'impression.

Remplaçons-le par une virgule, et trois espaces seront automatiquement insérés à la place du retour à la ligne.

Ces trois possibilités de l'instruction PRINT sont vitales lorsqu'il s'agit de "mettre en page" (ou plutôt en écran), les résultats d'un traitement informatique ou, tout simplement, dans le cadre des jeux vidéo.

Toutes ces modifications nous ont montré l'intérêt qu'il y a à numéroter les lignes de programme de dix en dix : on peut ainsi sans problème en intercaler de nouvelles en cas de besoin.

Ce petit programme nous montre également l'aptitude étonnante des ordinateurs pour les taches répétitives : tant qu'un CTRL C ne sera pas lancé, l'ORIC alignera imperturbablement des "ORAC", sans jamais se fatiguer.

Peut-être serait-il plus intéressant de fixer des limites à son ardeur ?

Pour ce faire, nous allons mettre à contribution quelques-unes des instructions les plus importantes du BASIC.

Imaginons donc que nous souhaitions aligner les uns au dessus des autres cinq mots, par exemple ORIC. Il est clair qu'il faut *compter* les impressions, et sortir de la boucle dès que cinq opérations ont été enregistrées.

Frappons donc le programme suivant :

```
10 PRINT "ORIC"
20 A=A+1
30 IF A<5 THEN 10
```

Au départ, la variable A contient la valeur zéro, mais il est bien entendu qu'une instruction antérieure aurait pu lui en affecter une autre (par exemple : 5 A = 2).

Après chaque impression, la ligne 20 ajoute une unité à la variable A. Insistons sur le fait que cette ligne ne prétend pas être une égalité au sens mathématique du terme. Son sens exact est : "soit" A = A + 1. On "écrase" l'ancienne valeur de A par le résultat de l'addition de cette ancienne valeur avec la constante 1.

Les puristes pourront, comme c'est obligatoire sur certaines autres machines, libeller ainsi la ligne 20 :

```
20 LET A = A + 1
```

Signalons à cette occasion qu'il est possible d'incorporer des espaces à l'intérieur des instructions pour en clarifier l'aspect. Ces espaces sont ignorés par la machine. Ainsi, la ligne 30 pourrait aussi bien être frappée de cette façon :

```
30IFA<5THEN10
```

ou de celle-ci :

```
30 IF A<5 THEN 10
```

de très loin plus agréable à déchiffrer !

Cette ligne 30 utilise l'une des possibilités les plus puissantes de l'informatique : le *branchement conditionnel.*

La partie d'instruction placée après THEN n'est exécutée que si la condition spécifiée après IF est réalisée. Dans le cas contraire, toute la ligne est ignorée, et l'exécution continue à la ligne suivante.

Ici, tant que A ne dépassera pas la valeur 5, la machine exécutera un GOTO 10, et bouclera donc. Notons qu'après THEN, il n'est pas obligatoire de frapper GOTO : le numéro de la ligne à aller exécuter suffit. Par contre, on peut omettre THEN si on conserve GOTO ! La ligne 30 pourrait ainsi prendre la forme suivante :

```
30IF A < 5 GOTO 10
```

Une fois le travail accompli, la machine s'arrête, puisqu'il n'y a pas de suite à notre programme. On aurait pu prévoir une ligne telle que :

```
40 PRINT : PRINT "OUF,TERMINE !"
```

Précisément, incorporons cette ligne à notre programme, et lançons-le de nouveau.

Nous pouvons constater que la machine exprime son soulagement seulement après avoir sauté une ligne.

En effet, la ligne 40 contient deux instructions distinctes séparées par deux points :

— PRINT et :
— PRINT "OUF,TERMINE !"

Un PRINT isolé n'imprime rien, mais déclenche un retour à la ligne immédiat, donc le saut d'une ligne.

Les deux points permettent de loger plusieurs instructions sur une même ligne, sans leur attribuer de numéros distincts. Pas question donc de prévoir ultérieurement un GOTO vers autre chose que le groupe complet d'instructions !

En fait, le procédé n'est réellement intéressant que lorsqu'il s'agit d'exécuter plusieurs instructions après un THEN.

Il évite de répéter la condition IF dans toutes les instructions conditionnelles.

Le programme suivant illustre cette possibilité tout en introduisant déjà une certaine mise en scène sonore, en avant-première de l'étude complète des possibilités sonores de la machine dans un prochain chapitre...

```
10 A=5
20 WAIT 100
25 PRINT A : PING
30 A=A-1
40 IF A=-1 THEN ZAP :WAIT 100:EXPLODE:STOP
50 GOTO 20
```

Il existe cependant une possibilité plus souple permettant d'accomplir une tâche donnée, un certain nombre de fois. Il s'agit de la *boucle FOR-NEXT*, autre outil très puissant offert par le langage BASIC.

Les lignes FOR F = 0 TO 5 et NEXT sont indissociables. Utiliser l'une sans l'autre déboucherait sur un compte rendu d'erreur du genre "NEXT WITHOUT FOR".

La première ligne signifie : exécuter les instructions qui vont suivre, pour toutes les valeurs de F comprises entre 0 et 5 incluses.

La seconde signifie "au suivant" et indique donc la fin de la séquence d'instructions. Lorsque NEXT est rencontré, la machine passe à la valeur suivante de la variable spécifiée après FOR, et revient au début de la séquence d'instructions, qui sera à nouveau

exécutée avec cette nouvelle valeur de F. Plusieurs boucles FOR-NEXT peuvent être imbriquées dans un même programme, mais il faut alors spécifier après NEXT le nom de la variable à faire progresser, afin d'éviter toute confusion : NEXT F.

```
10 FOR F=0 TO 5
20 PING
25 WAIT 100
30 PRINT 5-F
40 NEXT
50 ZAP
60 WAIT 100
70 EXPLODE
```

Sauf précision supplémentaire, la boucle FOR-NEXT ajoute une unité à la variable de boucle, lors de chaque NEXT. Il est facile de spécifier un autre pas, même négatif, ce qui ajoute encore à la souplesse du procédé :

```
10 FOR F=5 TO 0 STEP -1
20 PING
25 WAIT 100
30 PRINT F
40 NEXT
50 ZAP
60 WAIT 100
70 EXPLODE
```

Nous avons pu, jusqu'à présent, découvrir quelques-unes des principales aptitudes de l'ORIC :

— gestion de l'écran ;
— travaux répétitifs ;
— effets sonores ;
— manipulations de nombres et de textes.

Avant de passer à la découverte des possibilités de dialogue homme-machine, il nous faut exécuter une expérience plutôt édifiante, destinée à éviter (ou tout au moins à expliquer) certains comportements pour le moins surprenants de la machine.

Frappons donc le "programme" suivant :

```
10 ECRIS "ORIC"
20 AUGMENTE A
30 SI A<5 ALORS 10
```

Bien que de telles "instructions" soient tout à fait étrangères au BASIC (et c'est bien dommage !), l'ORIC ne refuse nullement de les mémoriser. Bien plus, il pousse la complaisance jusqu'à accepter de lister le tout...

Poussons plus loin la plaisanterie, et faisons RUN. Cette fois la mesure est comble et la machine signale une erreur de syntaxe dans la ligne 10, la toute première rencontrée ! L'ORIC n'examine donc la vraisemblance d'une ligne que lors de sa première tentative d'exécution, et pas avant, contrairement à d'autres ordinateurs (SINCLAIR par exemple), qui manifestent leur désaccord dès la fin de la frappe de la ligne.

Un surcroît d'attention est donc exigé de la part du programmeur, qui devra également contrôler *entièrement* le fonctionnement de son œuvre, avant de se déclarer satisfait : en effet, certains programmes possèdent des lignes qui ne s'exécutent que dans des cas très particuliers, donc peu souvent. Négliger de les "faire tourner" lors des essais comporte bien des risques, car certaines erreurs passent pour ainsi dire inaperçues au listage : une virgule au lieu d'un point dans une valeur numérique suffit amplement pour entraîner le refus d'une instruction....

Chapitre 2

Organisons le dialogue



Le dialogue entre la machine et son utilisateur est peut-être la fonction la plus importante d'un ordinateur.

Une forme de dialogue peut être reconnue dans l'écriture et la mise au point de programmes, mais celle-ci exige certaines connaissances de la part de l'opérateur.

Le véritable dialogue ne s'instaure cependant que lorsque la machine exécute un programme permettant à un utilisateur non programmeur de "converser" librement avec l'ordinateur, par l'intermédiaire de l'écran, du clavier, voire d'autres périphériques tels que des manettes de jeu.

Nous avons déjà examiné les possibilités offertes par l'écran, aussi allons-nous nous intéresser maintenant surtout au clavier.

L'instruction la plus classique servant à entrer des informations au clavier est INPUT. Deux variantes existent à son utilisation selon

que la machine "attend" une valeur numérique ou une chaîne de caractères (un texte).

Le programme que voici illustre ces diverses possibilités, tout en mettant l'écran à contribution.

```
10 CLS
20 PRINT"BONJOUR,JE M'APPELLE ORIC,"
30 PRINT"ET VOUS, QUEL EST VOTRE NOM ?"
40 INPUT A$
50 CLS
60 PRINT"HEUREUX DE VOUS CONNAITRE, ";A$
70 PRINT"QUEL AGE AVEZ VOUS ?"
80 INPUT A
90 CLS
100 PRINTA$;" A ";A;" ANS"
110 PRINT:PRINT"ET VOILA LE TRAVAIL !"
```

Lorsque la machine vous demande votre nom, vous pouvez répondre par n'importe quelle suite de caractères de moins de 79 éléments. Rien ne vous empêche de vous présenter par votre numéro de Sécurité Sociale !

Par contre, vous ne pourrez pas donner votre âge sous une forme autre qu'une valeur numérique.

Si vous tentez de répondre :

— NON

— CELA NE VOUS REGARDE PAS

— VINGT

— 20 ANS

la machine vous répondra "REDO FROM START" et vous donnera une nouvelle chance. Répondez par exemple 20 et tout ira bien !

Dans tous les cas, lorsqu'une instruction INPUT est exécutée, un point d'interrogation est inscrit sur l'écran, indiquant que la parole est à l'opérateur. Une fois la réponse frappée, il faut la *valider* en pressant RETURN. La parole est ainsi rendue à la machine qui, autrement, aurait bien du mal à savoir si votre réponse est complète ou non...

Dans bien des cas, cependant, la simplicité de la réponse à donner s'accomoderait fort bien de la simple pression sur une seule touche.

L'instruction GET est précisément prévue à cet effet.

```
10 CLS
20 PRINT "GARCON OU FILLE ?"
30 GET S$
40 IF S$="G" THEN CLS:GOTO 100
50 IF S$="F" THEN CLS:GOTO 200
60 GOTO 30
100 PRINT "BEAU OU LAID ?"
110 GET B$
120 IF B$="B" THEN CLS:PRINT
"QU'IL EST BEAU!":STOP
130 IF B$="L" THEN CLS:PRINT
"QU'IL EST LAID!":STOP
140 GOTO 110
200 PRINT "BELLE OU LAIDE ?"
210 GET B$
220 IF B$="B" THEN CLS:PRINT
"QU'ELLE EST BELLE!":STOP
230 IF B$="L" THEN CLS:PRINT
"QU'ELLE EST LAIDE!":STOP
240 GOTO 210
```

La principale différence avec INPUT est que GET ne lit qu'une seule touche. La machine met dans la chaîne A$ le caractère correspondant à la touche enfoncée lors de l'exécution de l'instruction GET, et ce, quel que soit ce caractère, puis le programme se poursuit.

On incorpore donc presque toujours les instructions GET à l'intérieur de boucles s'exécutant sans relâche tant qu'une touche prévue n'a pas été pressée.

Il n'y a bien entendu pas lieu de presser RETURN dans de tels cas, ce qui augmente sensiblement la "vivacité" du dialogue.

L'instruction GET peut aussi servir à l'acquisition de valeurs numériques :

```
10 CLS
20 PRINT:PRINT:PRINT"                         MENU"
30 PRINT"                    ----"
40 PRINT:PRINT:PRINT"OPTION 1"
```

Suite au verso

```
50 PRINT:PRINT"OPTION 2"
60 PRINT:PRINT"OPTION 3"
70 GET N
80 IF N=1 THEN CLS: GOTO 1000
90 IF N=2 THEN CLS: GOTO 2000
100 IF N=3 THEN CLS: GOTO 3000
110 GOTO 70
1000 PRINT"ICI L'OPTION 1"
1999 STOP
2000 PRINT"ICI L'OPTION 2"
2999 STOP
3000 PRINT"ICI L'OPTION 3"
```

Ce programme illustre l'utilité de GET lorsqu'il s'agit de sélectionner l'une des options proposées par ce qu'il est convenu d'appeler le "menu" d'un programme.

Cette technique sert aussi bien pour les applications professionnelles (comptabilité, traitement de textes, etc.) que dans le cadre de jeux vidéo (choix d'une force de jeu, sélection d'une variante, consultation des règles ou démarrage direct de la partie, etc.).

Remarquons que si la machine attend un chiffre (et un seulement), le fait de frapper une touche de lettre ou de signe bloque le programme sur une erreur de syntaxe, et non sur un simple "REDO FROM START" offrant une chance de rachat ! Sauf nécessité absolue, c'est donc plutôt sous la forme "acquisition de chaîne" que sera utilisée l'instruction GET. Il existe toutefois une instruction "concurrente" pour GET, utilisant le mot-clé KEY$ avec certains avantages.

```
5 PRINTCHR$(20)
10 CLS
20 GET A$
30 IF A$=CHR$(13) THEN PRINT :GOTO 20
40 IF A$=CHR$(127) THEN 10
50 IF A$=CHR$(27) THEN PRINT CHR$(20):STOP
60 PRINT A$;
70 GOTO 20
```

Ce programme permet d'utiliser l'ORIC exactement comme une machine à écrire, dont le papier ne serait autre que l'écran. Bien

sûr, les minuscules sont disponibles, et il faut actionner SHIFT pour "appeler" les capitales.

On constatera cependant deux inconvénients :

— le programme ne peut être arrêté par CTRL C ;

— l'apostrophe n'est pas reconnue.

Remplaçons la ligne 20 par :

20 A$ = KEY$

et tout rentre dans l'ordre. Le choix entre GET et KEY$ dépend, bien sûr, de l'application exacte envisagée, mais ces deux fonctions ouvrent bien des horizons !

Dans ce programme apparaît à plusieurs reprises le mot-clé CHR$. Ce mot autorise toute une variété d'applications dont voici deux exemples :

— ligne 5 PRINT CHR$ (20) ;

— ligne 40 IF A$ = CHR$ (127) THEN 10

Il faut d'abord savoir que, dans la machine, tout caractère *quel qu'il soit* (lettre, chiffre, signe, symbole), est traité sous la forme d'un *octet,* c'est-à-dire d'un nombre compris entre 0 et 255.

La correspondance entre tous les caractères que la machine est capable de traiter, et ces nombres s'appelle le code ASCII, résumé dans notre tableau.

32		33	!	34	"	35	#
36	$	37	%	38	&	39	'
40	(	41	)	42	*	43	+
44	,	45	-	46	.	47	/
48	0	49	1	50	2	51	3
52	4	53	5	54	6	55	7
56	8	57	9	58	:	59	;
60	<	61	=	62	>	63	?
64	@	65	A	66	B	67	C
68	D	69	E	70	F	71	G
72	H	73	I	74	J	75	K
76	L	77	M	78	N	79	O
80	P	81	Q	82	R	83	S
84	T	85	U	86	V	87	W
88	X	89	Y	90	Z	91	[

Suite au verso

92	\	93	]	94	^	95	£
96	©	97	a	98	b	99	c
100	d	101	e	102	f	103	g
104	h	105	i	106	j	107	k
108	l	109	m	110	n	111	o
112	p	113	q	114	r	115	s
116	t	117	u	118	v	119	w
120	x	121	y	122	z	123	{
124	|	125	}				

La lettre A, par exemple, correspond au code 65, et le mot CHR$ permet de mettre en pratique la correspondance :

PRINT CHR$ (65)

fera ainsi apparaître la lettre A sur l'écran.

Inversement, le mot ASC permet de passer du caractère à son code ASCII :

PRINT ASC ("A")

renvoie effectivement la valeur 65, alors que :

PRINT CHR$ (ASC("A"))

imprime bien évidemment un A.

Seulement, le jeu de caractères de la machine n'utilise que les codes 32 à 125. Parmi les codes restants, certains sont, bien sûr, inutilisés, mais d'autres servent de *"caractères de contrôle"*, soit pour l'ordinateur, soit pour une éventuelle imprimante.

Si on incorpore ces caractères de contrôle dans un ordre PRINT, rien ne sera imprimé, mais une modification interviendra dans le fonctionnement de l'ordinateur :

Ainsi, PRINT CHR$ (20) commande le blocage ou le déblocage des minuscules du clavier, exactement comme l'action sur les touches CTRL T. On remarquera d'ailleurs que T est la *vingtième* lettre de l'alphabet, ce qui n'est pas sans rapport...

Voici, en effet, la liste des codes de contrôle de l'ORIC :

TOUCHE avec CTRL	CODE CHR$	Effet de la commande
A	**1**	acquisition d'un caractère présent à l'écran
C	**3**	arrêt d'un programme qui « boucle »
D	**4**	dédoublement de lignes
F	**6**	suppression bruitage clavier
G	**7**	émission d'un son de clochette
H	**8**	curseur 1 cran à gauche
I	**9**	curseur 1 cran à droite
J	**10**	curseur 1 cran vers le bas
K	**11**	curseur 1 cran vers le haut
L	**12**	vidage d'écran (CLS)
M	**13**	retour chariot (à la ligne) RETURN
N	**14**	masquage de la ligne en cours de frappe
O	**15**	neutralise l'écran
P	**16**	neutralise l'imprimante
Q	**17**	fait disparaître le curseur
S	**19**	neutralise l'écran
T	**20**	bloquage/débloquage capitales
X	**24**	annulation de la ligne frappée
Z	**26**	identique à ESCAPE (CHR$(27))
[	**27**	ESCAPE
]	**29**	libération colonne gauche écran

Finalement, la commande par la touche CTRL permet une action manuelle sur le système (en mode commande), alors que le passage par un ordre PRINT CHR$ permet les mêmes actions pendant le déroulement d'un programme (la machine agit elle-même sur son fonctionnement).

Voici donc la raison d'être de la ligne 5, qui permet à l'utilisateur de disposer des lettres minuscules.

Dans la ligne 40, le problème est un peu différent : il existe, en effet, quelques touches dont l'effet est le même que certaines des combinaisons CTRL qui viennent d'être vues. Ces touches sont RETURN, ESCAPE, et les quatre touches fléchées. Ces touches servant très souvent, l'utilisateur appréciera de n'avoir plus à presser CTRL puis une seconde touche.

Ajoutons enfin la touche DEL qui joue un rôle spécial et dont le code ASCII est 127.

Voici les codes de ces touches "pas comme les autres" :

ESCape	27
DEL	127
RETURN	13
BARRE ESPACE	32
←	8
→	9
↓	10
↑	11

Après un GET A$ il est bien sûr facile de reconnaître l'appui sur A en programmant :

IF A$ = "A" THEN....

En revanche, il n'est pas possible d'écrire :

IF A$ = "DEL" THEN....

On utilise donc la formulation suivante :

IF A$ = CHR$ (127) THEN.....

et le tour est joué, l'appui sur DEL pourra être détecté !

Notons que rien n'empêche de remplacer :

IF A$ = "A" par IF A$ = CHR$ (65)

mais l'intérêt de la manœuvre n'est guère évident...

Revenons donc à notre programme de la "machine à écrire" : l'appui sur RETURN entraîne le retour à la ligne, ESCape arrête le programme (puisque CTRL C est inopérant avec GET), alors qu'en pressant DEL, on efface tout et on recommence ! Une amélioration consisterait à remplacer la ligne 20 par :

20 A$ = KEY$

Ainsi, CTRL C deviendrait utilisable, mais surtout l'apostrophe pourrait s'imprimer.

Nous venons donc de prendre connaissance des diverses possibilités permettant l'échange d'informations entre l'opérateur et la machine.

L'ORIC peut cependant aussi dialoguer avec un magnétophone à cassettes (ou à bobines, pourquoi pas ?), ce qui est précieux lorsqu'il s'agit de mettre un programme en lieu sûr avant d'arrêter l'ordinateur.

Une prise DIN spéciale à sept broches sert à la fois au raccordement du magnétophone et de la chaîne HIFI, s'il y a lieu.

Mise en œuvre d'un magnétophone

La très haute qualité de l'interface cassette est l'un des points forts de l'ORIC.

Les transfuges de machines telles que le ZX-81 seront émerveillés par l'aptitude de l'ORIC à s'accomoder des pires cassettes insérées dans les pires magnétophones avec les pires réglages. Bien sûr, l'usage de matériel de qualité raisonnable reste conseillé car même un ORIC éprouvera de sérieuses difficultés à relire une bande qui aura "bouchonné" derrière un cabestan...

Le seul point faible de l'interface cassette réside dans le brochage de la prise DIN, aussi peu standard que ne l'est la conduite à gauche en vigueur au pays des ORIC.

Il faudra donc réaliser un cordon permettant de relier la broche 1 de la prise DIN à l'entrée micro du magnétophone, la broche 3 à sa sortie écouteur, et la masse de l'enregistreur à la broche 2, par l'intermédiaire du blindage des câbles de liaison.

Les broches 6 et 7 de la prise DIN rejoignent le contact travail d'un petit relais situé dans l'ORIC, et peuvent être reliées, sans sens préférentiel, aux deux contacts de la prise de télécommande du magnétophone. Ainsi, le moteur ne tournera que lorsque cela est nécessaire.

En pratique, cette facilité n'est pas vraiment commode, puisqu'il faut débrancher la prise de télécommande dès que des manipulations de recherche sont nécessaires (avance ou retour rapides).

De plus, la consommation des moteurs de certains magnétophones est supérieure aux possibilités du contact du relais, qui risque parfois de se souder.

Pour sauvegarder un programme, il suffit de frapper :

CSAVE "NOM DU PROGRAMME"

et de presser RETURN dès que la bande a démarré.

Une tonalité de pré-réglage du niveau d'enregistrement est émise, puis le programme est transféré. Tout va très vite, comme en témoigne le peu de temps pendant lequel le texte "saving NOM DU PROGRAMME" subsiste en haut de l'écran. Par sécurité (un défaut dans la bande est toujours possible), on peut déclencher une seconde sauvegarde en utilisant la procédure suivante :

Au moyen de la touche à flèche verticale, amener le curseur au

début de la ligne "CSAVE". Presser alors CTRL A jusqu'à ce que toute la ligne ait été balayée, appuyer sur RETURN, et c'est reparti !

Lorsqu'une fiabilité extrême est indispensable, on peut ralentir le transfert en frappant :

CSAVE "NOM DU PROGRAMME",S

Toutefois, il nous semble à la fois plus sûr et plus rapide de doubler ou même tripler une sauvegarde rapide. A vous de choisir !

En revanche, les cassettes du commerce sont presque toujours enregistrées en mode lent, car les enregistrements rapides supportent mal la duplication industrielle.

Pour le premier chargement en machine, il convient donc de faire :

CLOAD "NOM DU PROGRAMME",S

ou simplement :

CLOAD " ",S

La touche RETURN pourra être pressée avant même que le magnétophone ne soit mis en route. L'indication "Searching" apparaîtra en haut de l'écran, et se transformera en "Loading" dès que le chargement sera en cours.

Si on souhaite interrompre le chargement en cours de route, l'action sur le poussoir RESET, camouflé sous la machine, est à peu près la seule solution.

Les cassettes du commerce étant généralement enregistrées sur leurs deux faces, nous ne saurions trop recommander à nos lecteurs de "récupérer" l'une des pistes pour y loger quelques copies accélérées, pour le confort des futures utilisations. Pour ce faire, il suffit d'obturer au ruban adhésif, l'encoche de protection de la piste à surcharger. Lors de la sauvegarde d'un programme sur cassette, on peut demander à ce que la machine exécute automatiquement un RUN dès la fin du rechargement. Il faut pour cela libeller ainsi la commande de sauvegarde :

CSAVE "NOM DU PROGRAMME",AUTO

Les autres fonctions de sauvegarde (notamment d'écrans) fonctionnent de façon plus ou moins défectueuse sur l'ORIC 1, et devront être donc souvent évitées. Les possesseurs d'ATMOS, pourront se montrer plus hardis, et disposeront même de possibilités supplémentaires telles que STORE, RECALL, JOIN et VERIFY qui, bien que secondaires, peuvent s'avérer confortables.

Chapitre 3

L'ORIC en vedette

Dialoguer avec l'ordinateur est une chose, mais encore faut-il apprendre à la machine à traiter les informations qui lui sont soumises pour arriver à un résultat de quelque utilité.

Dans ce chapitre, nous allons présenter quelques exemples de programmes utilisant les principales fonctions de base du langage BASIC pour résoudre, dans les meilleurs délais, des problèmes courants.

Un peu de comptabilité

Commençons par un court programme mettant à la fois à contribution les possibilités de calcul numérique de l'ORIC, et ses aptitudes en matière d'édition des résultats.

Il suffira bien évidemment de reprendre les principes utilisés ici pour mettre sur pied, par simple modification des formules utilisées, toute une variété de programmes de calcul numérique.

```
5 CLS
10 PRINT "MONTANT HT ?"
20 INPUT HT
25 CLS
30 PRINT:PRINT:PRINT
40 PRINT "                    FACTURE"
50 PRINT "                    -------"
60 PRINT:PRINT:PRINT:PRINT:PRINT
70 PRINT "Montant HT.........";HT
72 TVA=(INT(0.01+HT*18.6))/100
75 PRINT "TVA 18,60 % ........";TVA
80 PRINT:PRINT: PRINT "
    -----------"
90 PRINT:PRINT "TOTAL TTC..........."
;HT+TVA;" F"
```

Nous voici donc plongés dans le domaine de la comptabilité ! Ne nous y trompons pas, malgré son extrême simplicité, ce petit programme est tout à fait sérieux : transposé sur une imprimante "papier ordinaire" en lieu et place de l'écran, complété par quelques lignes permettant de "personnaliser" les documents établis, il serait capable de rendre d'appréciables services à une petite entreprise !

Nous connaissons des professionnels qui, grâce au système ORIC, ne passent pas plus de trente secondes pour établir une facture en bonne et due forme...

Vers la "facturation détaillée" ?

Nous avons mis au point ce petit programme dans le but d'aider les abonnés dont les factures de téléphone grèvent exagérément le budget.

Il faut bien reconnaître, en effet, que la prolongation incontrôlée de communications interurbaines risque d'occasionner des frais dont il n'est pas toujours aisé d'apprécier l'importance.

Facturation détaillée et compteur à domicile sont des services coûteux qu'un ordinateur domestique peut au moins partiellement remplacer.

```
40 CLS:P=12
50 PRINT:PRINT:PRINT  "TARIF REDUIT ? O/N"
60 GET A$
70 IF A$="O" THEN P=2*P : GOTO 100
80 IF A$="N" THEN 100
90 GOTO 60
100 CLS
110 PRINT:PRINT:PRINT "APPUYEZ SUR UNE TOUCHE"
120 PRINT:PRINT "DES LE DECROCHAGE DU DEMANDE"
130 GET A$ : PING
140 T=0.60
150 CLS
160 PRINT"DEJA ";T;" F"
170 L=P
180 FOR N=1 TO L
190 PLOT N,20,"*"
200 NEXT
210 WAIT100
220 L=L-1
230 PLOT L,20,CHR$(32)
240 IF L=1 THEN 260
250 GOTO 210
260 T=T+0.60: PING :WAIT 60
270 GOTO150
```

Le service rendu par ce programme est triple :

• émission d'un signal sonore lors de chaque imputation d'une taxe de base au compteur ;
• affichage permanent en francs et centimes de ce qu'a déjà coûté l'appel ;
• animation matérialisant le "grignotage" de la taxe de base en cours, permettant de raccrocher au moment le plus avantageux.

Pour des raisons de simplicité et de précision, il a été décidé de limiter le domaine d'application du programme aux seules communications taxées au tarif d'une unité toutes les douze secondes (en période de plein tarif).

Cela correspond à la majorité des appels interurbains pouvant être établis depuis un point donné de l'hexagone.

Il serait cependant très possible de modifier le programme pour lui faire prendre en compte d'autres périodicités de taxation.

Là n'est toutefois pas, à notre avis, le rôle de ce programme, qui ne prétend en aucun cas se substituer au compteur officiel mais, plus modestement, aider le consommateur à prendre conscience du coût prohibitif de certains appels. Un dialogue très simple permet à la machine de prendre en compte le tarif réduit lorsqu'il est applicable (voir l'annuaire), et ainsi d'en démontrer l'intérêt de façon tangible. Les lignes 140 et 260 utilisent la valeur de la taxe de base, et devront bien sûr être remises à jour lorsque celle-ci augmentera au-delà de soixante centimes (ou diminuera...). Ce programme peut facilement être interrompu dans son déroulement par action sur CTRL C.

Un peu de distraction !

L'un des domaines privilégiés d'utilisation des ordinateurs domestiques est sans nul doute le secteur des jeux vidéo. La mise au point d'un bon programme de jeu utilisant toutes les ressources d'une machine (son, couleur, haute résolution graphique, etc.) est un très gros travail tout à fait hors de portée du débutant, ce qui explique le succès des cassettes pré-enregistrées disponibles dans le commerce.

Nous allons cependant lever le voile sur les principes de base utilisés dans les jeux d'animation.

```
5 CLS
10 X=1+(RND(1)*37)
20 Y=RND(1)*26
30 PLOT X,Y,CHR$(96)
40 X=19
50 Y=13
60 PLOT X,Y,CHR$(43)
65 A=X : B=Y
70 GET A$
80 IF A$= CHR$(8) THEN X=X-1
90 IF A$= CHR$(9) THEN X=X+1
```

```
100 IF A$= CHR$(10) THEN Y=Y+1
110 IF A$= CHR$(11) THEN Y=Y-1
120 IF X<1 THEN X=1
130 IF X>38 THEN X=38
140 IF Y<0 THEN Y=0
150 IF Y>26 THEN Y=26
160 IF SCRN(X,Y)=96 THEN 200
165 PLOT A,B,CHR$(32)
170 GOTO 60
200 PLOT X,Y,"?"
210 EXPLODE
```

Ce petit programme n'est pas à proprement parler un jeu car sa "mise en scène" est réduite au strict nécessaire.

Le lecteur trouvera dans les chapitres consacrés au son et aux graphismes tout un choix d'idées permettant de "planter le décor" avec tout le raffinement souhaitable. Quant aux méthodes de calcul numérique qui ont été exposées, elles pourront trouver une utilité lorsqu'il sera nécessaire d'afficher des scores constamment remis à jour au fil de la partie.

Le tout premier soin de ce programme de démonstration est de placer un signe © en un point de l'écran *choisi au hasard.* Le rôle du hasard est déterminant dans la plupart des jeux, aussi l'existence de la fonction aléatoire RND est-elle indispensable sur un ordinateur domestique.

Sur l'ORIC, RND(1) est un nombre fractionnaire quelconque compris entre 0 et 1. Des opérations mathématiques simples permettent de transformer ce nombre selon les besoins du programme utilisateur.

Par exemple, la ligne 10 élabore un nombre aléatoire compris entre 1 et 38, alors que la ligne 20 en construit un autre, compris entre 0 et 26.

Pourquoi ces limites ? Tout simplement parce qu'elles correspondent au domaine accessible à la fonction PLOT de la ligne 30. Le non-respect de ces limites entraînerait très vite un blocage du programme.

La ligne 60, pour sa part, place un signe + au milieu de l'écran, mais le "joueur" pourra déplacer cette "arme" au moyen des touches fléchées du clavier.

Ce sont les lignes 80 à 110 qui prennent en charge les déplacements proprement dits, alors que les lignes 120 à 150 imposent à

ces mouvements les limites déjà évoquées. Il pourra d'ailleurs être instructif de supprimer ces diverses "sécurités" et de juger de l'effet produit...

Avec la ligne 160, nous découvrons une possibilité très puissante de l'ORIC. L'instruction SCRN (screen) permet en effet de "lire" tout point de l'écran, puisqu'elle donne accès *au code* du caractère situé au point interrogé.

Ce point est repéré au moyen de coordonnées identiques à celles utilisées par l'instruction PLOT.

La ligne 165 sert à éviter que les déplacements de l'arme ne laissent un sillage (essayons de l'omettre !), alors que les lignes 200 et 210 mettent en scène la fin tragique de la cible...

Les plus curieux de nos lecteurs pourront chercher à comprendre le rôle de la ligne 65, ainsi que du remplacement de X et Y par A et B dans la ligne 165 !

Les petits secrets du tri postal

Depuis que le code postal est entré dans les mœurs, on remarque très souvent une rangée de petits bâtonnets rose-orangé dans le bas des enveloppes remises par le facteur. Curieusement, un trait de stylo-bille vient généralement rayer ces hiéroglyphes sur les lettres qui semblent avoir "traîné en route"...

En fait, dans les centres de tri, ces petits bâtonnets fluorescents (appelés marques d'indexation), sont apposés par une machine, à partir d'une lecture du code postal indiqué par l'expéditeur.

Par la suite, la lettre sera aiguillée de machine en machine uniquement par lecture automatique de ces marques. L'avantage est une énorme accélération des procédures de tri. Au chapitre des inconvénients, il faut savoir qu'en cas de codage défectueux (suite, par exemple à une adresse peu lisible), la lettre sera systématiquement acheminée dans une mauvaise direction, tant que les marques d'indexation n'auront pas été rayées manuellement.

```
5 CLS
10 PRINT:PRINT:PRINT "DONNER UN NUMERO
DE CODE POSTAL"
20 INPUT Z$
30 PRINT:PRINT:PRINT Z$
40 PRINT:PRINT
```

```
50 FOR I=1 TO 5
60 Y$= MID$(Z$,I,1)
70 A=VAL(Y$)
80 IF A=0 THEN B$=B$ + "||||  "
90 IF A=1 THEN B$=B$ + "||| | "
100 IF A=2 THEN B$=B$ + "|| || "
110 IF A=3 THEN B$=B$ + "| ||| "
120 IF A=4 THEN B$=B$ + "|||  |"
130 IF A=5 THEN B$=B$ + "|| | |"
140 IF A=6 THEN B$=B$ + "| || |"
150 IF A=7 THEN B$=B$ + "||  ||"
160 IF A=8 THEN B$=B$ + "| | ||"
170 IF A=9 THEN B$=B$ + "|  |||"
180 NEXT
190 PRINT"VOICI LES MARQUES D'INDEXATION:"
200 PRINT:PRINT:PRINT
210 FOR L=LEN(B$) TO 1 STEP -1
220 PRINT MID$(B$,L,1);
230 NEXT
240 PRINT:PRINT:PRINT:PRINT
250 PRINT "UN AUTRE CODE ? O/N"
260 GET X$
270 IF X$="O" THEN RUN
280 GOTO 260
300 REM COPYRIGHT 1983
```

Ce programme permet de reconstituer, sur l'écran, une réplique exacte des marques devant correspondre à un code postal à cinq chiffres entré au clavier.

On pourra ainsi retracer le chemin parcouru par les lettres ayant subi une indexation défectueuse, ou... étonner ses amis et connaissances ou, pourquoi pas, son facteur !

Nous découvrons ici des instructions nouvelles du BASIC : la fonction MID$, qui permet de prélever des caractères à un endroit spécifié d'une chaîne. Ainsi, MID$(A$,5,3) représenterait les trois caractères pris à partir du cinquième de A$. A la ligne 60 de notre programme, on se contente simplement de prélever le I^{ième} caractère.

A la ligne 70, nous remarquons la fonction VAL(Y$).

VAL extrait la VALeur numérique d'une chaîne, à condition bien sûr

que celle-ci soit composée uniquement de chiffres ! Ainsi, on peut simplifier la suite du programme en remplaçant IF Y$ = "0" par IF A = 0, et ainsi de suite jusqu'à 9. Encore une instruction de traitement de chaînes avec LEN. LEN(B$) représente tout simplement le nombre de caractères de la chaîne B$, autrement dit sa *longueur* (LENgth).

L'ORIC, opérateur radiotélégraphiste

Avec ce programme, nous abordons les applications spectaculaires de l'informatique individuelle, sans pour autant mettre à contribution des trésors d'ingéniosité.

```
10 GOTO 200
20 CLS
25 FOR F=1 TO LEN(A$)
30 IF MID$(A$,F,1)="\"THEN GOSUB 150
40 IF MID$(A$,F,1)="\"THEN 60
50 IF MID$(A$,F,1)="."THEN L=5
55 IF MID$(A$,F,1)="-"THEN L=15
57 GOSUB 100
60 NEXT
70 PRINT "MEME MESSAGE ? O/N"
72 GET K$
75 IF K$="O" THEN 20
80 IF K$="N" THEN 200
85 GOTO 72
100 SOUND 1,100,13
120 WAIT L
125 SOUND 1,0,1
130 RETURN
150 WAIT 10
160 RETURN
200 A$=""
205 CLS
210 PRINT "FRAPPEZ VOTRE MESSAGE"
220 PRINT:PRINT "      PUIS RETURN"
```

```
230 INPUT B$
250 CLS
260 PRINT:PRINT:PRINT "TRADUCTION MORSE
EN COURS"
270 FOR F=1 TO LEN(B$)
280 C$= MID$(B$,F,1)
290 GOSUB 300
292 A$=A$+D$
295 NEXT
296 CLS
298 GOTO 20
300 IF C$="A" THEN D$=".-\"
310 IF C$="B" THEN D$="-...\"
320 IF C$="C" THEN D$="-.-.\"
330 IF C$="D" THEN D$="-..\"
340 IF C$="E" THEN D$=".\"
350 IF C$="F" THEN D$="..-.\"
360 IF C$="G" THEN D$="--.\"
370 IF C$="H" THEN D$="....\"
380 IF C$="I" THEN D$="..\"
390 IF C$="J" THEN D$=".---\"
400 IF C$="K" THEN D$="-.-\"
410 IF C$="L" THEN D$=".-..\"
420 IF C$="M" THEN D$="--\"
430 IF C$="N" THEN D$="-.\"
440 IF C$="O" THEN D$="---\"
450 IF C$="P" THEN D$=".--.\"
460 IF C$="Q" THEN D$="--.-\"
470 IF C$="R" THEN D$=".-.\"
480 IF C$="S" THEN D$="...\"
490 IF C$="T" THEN D$="-\"
500 IF C$="U" THEN D$="..- \"
510 IF C$="V" THEN D$="...-\"
520 IF C$="W" THEN D$=".--\"
530 IF C$="X" THEN D$="-..-\"
540 IF C$="Y" THEN D$="-.--\"
550 IF C$="Z" THEN D$="--..\"
560 IF C$="." THEN D$=".-.-.-\"
```

Suite au verso

```
580 IF C$="," THEN D$="--..--\"
590 IF C$="1" THEN D$=".----\"
600 IF C$="2" THEN D$="..---\"
610 IF C$="3" THEN D$="...--\"
620 IF C$="4" THEN D$="....-\"
630 IF C$="5" THEN D$=".....\"
640 IF C$="6" THEN D$="-....\"
650 IF C$="7" THEN D$="--...\"
660 IF C$="8" THEN D$="---..\"
670 IF C$="9" THEN D$="----.\"
680 IF C$="0" THEN D$="-----\"
690 IF C$=" " THEN D$="\"
900 RETURN
1000 REM COPYRIGHT 1983
```

La longueur de ce programme est uniquement due à l'existence d'une grande table de transcodage, puisque le but de la manœuvre consiste à faire émettre, au haut-parleur de l'ORIC, les signaux morse correspondant à un texte frappé au clavier. Le reste du programme ne fait appel qu'à une seule instruction nouvelle, le couple GOSUB/RETURN.

Nous disons bien *une seule* instruction, puisque GOSUB et RETURN ne peuvent être utilisés séparément (le message d'erreur RETURN WITHOUT GOSUB étant là à cet effet !).

GOSUB s'emploie de façon analogue à GOTO, mais essentiellement dans les cas où le *sous-programme* ainsi appelé sert de multiples fois. On le fait alors se terminer par RETURN, ce qui permet à la machine de revenir d'elle-même exécuter l'instruction *suivant immédiatement le GOSUB,* dès le terme du sous-programme. De substantielles économies d'espace mémoire peuvent ainsi être obtenues.

L'utilisation du programme est très simple, mais doit tenir compte de la limite imposée par l'ORIC en matière de longueur de chaînes : les messages trop longs risquant d'être tronqués, on aura intérêt à ne traiter que des mots isolés, ou des groupes courts. On pourrait même modifier très facilement le programme pour transmettre chaque caractère dès sa frappe.

Le signal sonore est disponible sur haut-parleur, mais également sur la broche 4 ou 5 de la prise magnétophone (la masse étant reliée à la broche 2).

Bien qu'il soit possible de piloter directement un émetteur avec ce signal, il nous semble que c'est surtout en matière d'enseignement

de la graphie que ce programme offrira un réel intérêt. En effet, le morse s'apprend en retenant des ensembles caractéristiques de sonorités, et non en séparant les points des traits. L'élève pourra entendre immédiatement les sons correspondant aux mots qu'il aura lui-même choisis, ce qui présente un certain intérêt pédagogique.

La répétition étant, dit-on, à la base de l'enseignement, il a été prévu une facilité permettant d'écouter autant de fois que nécessaire le dernier message émis.

L'instruction SOUND, qui apparaît deux fois dans ce programme, sera étudiée en détail dans le chapitre consacré aux possibilités sonores de l'ORIC, aussi ne nous étendrons-nous pas ici sur cette question.

L'ORIC extra-lucide ?

Un ordinateur peut exécuter très rapidement des calculs relativement complexes, du genre de ceux que seuls les illusionnistes arrivent à effectuer de tête sans effort décelable.

Dès lors, en programmant en machine certaines formules employées par les "magiciens", peut-on obtenir des résultats assez déconcertants pour les non-initiés.

```
1 CLS:PRINT:PRINT:PRINT"CHOISISSEZ UN NOMBRE":
A=6
2 PRINT"DE CINQ CHIFFRES"
3 GOSUB 100
4 PRINT:PRINT:PRINT"MULTIPLIEZ CE NOMBRE
PAR 2"
5 GOSUB 100
6 PRINT:PRINT:PRINT"AJOUTEZ 5"
7 GOSUB 100
8 PRINT:PRINT:PRINT"MULTIPLIEZ PAR 50"
9 GOSUB 100
10 PRINT:PRINT:PRINT"ADDITIONNEZ VOTRE AGE"
11 GOSUB 100
12 PRINT:PRINT:PRINT"QUEL EST LE RESULTAT
OBTENU ?"
```

Suite au verso

```
13 INPUT N
14 CLS
15 N=N-250
16 N$=STR$(N)
17 PRINT:PRINT:PRINT"VOUS AVIEZ CHOISI ";
LEFT$(N$,A)
18 PRINT:PRINT:PRINT"ET VOUS AVEZ ";
RIGHT$(N$,(LEN(N$)-A));" ANS"
20 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:STOP
100 PRINT:PRINT:PRINT "PUIS PRESSEZ UNE TOUCHE"
110 GET A$
115 CLS
120 RETURN
```

Nous n'entrerons pas dans les détails arithmétiques de ce programme, car leur niveau est voisin de celui du certificat d'études.

En revanche, il nous faut commenter encore trois fonctions de manipulation de chaînes :

LEFT$(N$,A), qui apparaît à la ligne 17, ne représente rien d'autre que les A, premiers caractères de N$.

Le cas de RIGHT$(N$,B) est très semblable, puisqu'il s'agit des B, *derniers* caractères de N$.

Dans ces deux exemples, A et B peuvent être remplacés par des expressions complexes qui seront évaluées au passage (comme dans la ligne 18).

STR$(N) appelle davantage de commentaires : cette fonction construit une chaîne à partir d'une valeur numérique. C'est donc normalement le contraire exact de VAL.

L'ORIC 1 possède une façon étrange d'exécuter STR$! Il ajoute un CHR$(2) en tête de la chaîne construite. A l'impression, ce défaut n'apparaît pas, puisque CHR$(2) n'est pas un caractère imprimable. Par contre, l'application de LEN après STR$ donne un résultat trop grand d'une unité.

La ligne 1 de notre programme tient compte de ce défaut, mais les possesseurs de machines ATMOS auront à remplacer A=6 par A=5.

A ce détail près, tout le reste du programme n'est qu'un dialogue utilisant des méthodes bien connues.

Chapitre 4

L'ORIC ingénieur du son

Jusqu'à présent, nous avons mis en œuvre des fonctions assez classiques du langage BASIC, disponibles sur à peu près tous les ordinateurs individuels. L'ORIC ne se distingue ici de ses rivaux que par des particularités qu'il faut connaître, mais qui n'ont rien de fondamental.

Parmi les vastes possibilités de l'ORIC, l'une des plus spectaculaires est, sans nul doute, la synthèse programmée de sons complexes. L'exploitation du générateur semble fort délicate au premier abord, mais devient très rapidement un jeu d'enfant !

Prise de contact

Il suffit de mettre l'ORIC sous tension pour s'apercevoir qu'il ne demande qu'à émettre des sons : un léger bruit de fond s'échappe du haut-parleur incorporé, et dès qu'une touche est pressée, un

petit “clic” est généré. Ce discret bruitage est fort utile en cas de frappe rapide (la qualité des touches mécaniques permet de très honnêtes performances), sans être notoirement énervant comme certains “bips” par trop répandus sur d'autres machines.

Les inconditionnels du clavier muet pourront, de toute façon, éliminer cette fonction en enfonçant la touche CTRL puis, sans la relâcher, la touche F. Il suffira de renouveler l'opération pour rendre la voix au clavier ainsi baillonné !

Une autre combinaison “de contrôle” (CTRL G) permet de provoquer manuellement l'émission d'un son de clochette. Le même résultat peut d'ailleurs être obtenu en utilisant l'ordre “PING” du BASIC. Chaque fois que sera lancée la commande PING (validée par RETURN), ce bruitage sera synthétisé. En mode programme, il faut se montrer un peu plus prudent.

Si nous lançons par RUN les deux lignes suivantes :

10 PING   20 GOTO 10

nous n'obtenons qu'un son continu (qu'il est facile d'arrêter par un CTRL C), car un nouveau PING est demandé bien avant que le précédent ne soit terminé.

En effet, *ce n'est pas* le microprocesseur 6502 qui élabore les sons, mais un circuit spécialisé AY-3-8912. Le microprocesseur se contente d'envoyer au synthétiseur un bref message décrivant le son à émettre, puis laisse le circuit spécialisé se charger de l'exécution, tout en vaquant à d'autres occupations. Cependant, si un nouveau message d'ordres arrive pendant l'exécution de l'ordre précédent, celui-ci est immédiatement interrompu au profit du nouveau.

Ici, l'ordre correspondant à PING comporte la description d'une diminution progressive de l'amplitude de la tonalité, mais un nouvel ordre est donné avant même que cette amplitude n'ait seulement commencé à décroître !

Ajoutons la ligne suivante : 15 WAIT 100

et le programme relancé nous plongera aussitôt dans l'ambiance d'un jeu radiophonique bien connu...

Des manipulations similaires peuvent être menées au moyen des autres *sons préprogrammés* que sont ZAP, SHOOT, et EXPLODE. Il s'agit là de bruitages brefs, choisis en fonction de leur fréquence d'apparition dans les jeux les plus courants.

Là ne s'arrêtent heureusement pas les possibilités sonores de l'ORIC !

Les instructions sonores

Le BASIC de l'ORIC dispose de trois mots-clé destinés à la construction d'instructions "sonores" : SOUND, MUSIC et PLAY. SOUND permet de définir des effets sonores très divers, mais pas forcément harmonieux (essentiellement des *bruitages).*

MUSIC permet d'obtenir très facilement les notes de la gamme grâce à un codage fort simple de demi-ton en demi-ton sur sept octaves.

PLAY commande à la fois l'*aiguillage* vers la sortie des divers sons pouvant être mixés entre eux, et la forme de l'*enveloppe* des signaux émis (en d'autres termes, les modalités de croissance et de décroissance de l'amplitude). C'est donc PLAY qui autorise la création d'effets de "percussion" ou de "piston". Normalement, un ordre PLAY devrait être associé à tout ordre SOUND ou MUSIC, et PLAY ne devrait pas pouvoir être utilisé seul.

En fait, une "initialisation" astucieuse permet d'obtenir très facilement des effets déjà intéressants :

Frappons la *commande :*

MUSIC 1,3,10,15

puis RETURN. Aussitôt apparaît une forte tonalité de 444 Hz, ce qui correspond au LA 3 à mieux que 1 % près.

Essayons de comprendre :

Le mot-clé MUSIC doit être suivi de quatre nombres (ou variables préalablement affectées) séparés par des virgules, et compris dans des fourchettes bien précises :

En première position doit figurer un chiffre 1, 2 ou 3 indiquant sur laquelle des trois voies sonores disponibles sera exécuté l'ordre MUSIC. Ce choix doit être cohérent avec celui mentionné dans l'ordre PLAY qui suit généralement. En l'absence d'ordre PLAY, le seul choix possible est celui de la voie 1 car les deux autres sont neutralisées.

En seconde position doit figurer le numéro de l'octave souhaitée, de Ø à 6. Le fait que la première octave soit numérotée Ø pourra quelque peu dérouter les musiciens, mais il sera toujours facile de s'accomoder de ce décalage d'une unité.

Le troisième nombre sera compris entre 1 et 12 et fixera la note émise, selon le code suivant :

1 DO	5 MI	9 SOL #
2 DO #	6 FA	10 LA
3 RE	7 FA #	11 LA #
4 RE #	8 SOL	12 SI

Enfin, on placera en quatrième position un nombre compris entre 1 et 15 qui déterminera le volume sonore du son émis : 1 correspond à un murmure à peine audible, alors que 15 est vraiment très fort !

L'indication 0 sera utilisée chaque fois que l'on souhaitera imposer une "enveloppe" au moyen d'un ordre PLAY : c'est alors le "générateur d'enveloppe" qui fera évoluer automatiquement l'amplitude de la note, sans possibilité de réglage manuel.

L'ordre SOUND fonctionne de façon assez similaire, avec les différences suivantes :

En première position, en plus des numéros des voies 1, 2, et 3, on dispose des possibilités 4, 5, et 6. 4 met en service la voie 1, 5 la voie 2, 6 la voie 3, mais en ajoutant un *bruit de souffle* à la tonalité. Cette possibilité n'est toutefois utilisable qu'en association avec PLAY.

En seconde position, un seul nombre compris entre 0 et 4095 fixera la hauteur (ou fréquence) du son émis. On se méfiera du fait que les valeurs 0, 1, 2, et 3 correspondent à des sons trop aigus pour être audibles !

En troisième position, on trouve le code fixant l'amplitude, comme précédemment (1 à 15, 0 en mode enveloppe).

Il n'y a pas de quatrième donnée dans un ordre SOUND.

Le court programme suivant va permettre de prendre connaissance de toute la gamme des tonalités disponibles :

```
10 FOR F=0 TO 4095
20 SOUND 1,F,15
30 NEXT
40 EXPLODE
```

Quelques remarques :

Premièrement, on rappelle qu'il n'est pas nécessaire de spécifier un nom de variable après NEXT lorsqu'il n'y en a qu'une. Ensuite, il est très net que la hauteur du son varie très vite dans les aigus, mais très, très lentement dans les graves. Cela provient du système de division de fréquence utilisé par le AY-3-8912, et signifie que les

fréquences basses seront produites avec une précision nettement meilleure que les fréquences très aiguës. En musique, cette remarque peut s'avérer importante, mais en bruitage ce n'est guère gênant.

Enfin, ce n'est pas par goût du spectacle que nous avons prévu la ligne 40 ! Supprimons-la et constatons que le système reste bloqué sur la tonalité la plus grave pouvant être synthétisée : le générateur de son attend un ordre et, à défaut, continue imperturbablement à exécuter le précédent.

En pratique, nous verrons qu'un PLAY 0,0,0,0 est le meilleur moyen d'arrêter le synthétiseur.

Remplaçons la valeur 4095 de la ligne 10 par 500, et nous obtiendrons (en présence d'EXPLODE) un effet intéressant !

Etudions enfin la fonction PLAY, cœur du synthétiseur ! L'utilisation de PLAY seul ne signifie normalement rien de précis. Il faut cependant savoir que le générateur de bruit de souffle est disponible en permanence et, qu'à défaut d'ordre contraire, les générateurs de tonalité se règlent sur une fréquence moyenne, soit environ 1 300 Hz.

Essayons de lancer la *commande* suivante par RETURN :

PLAY 7,0,4,300

Agaçant, n'est-ce pas ? Heureusement, une simple pression sur la barre d'espace (ou toute autre touche) arrête tout !

Voulez-vous recommencer ? Comme la commande est toujours présente sur l'écran amenez le curseur *juste devant* cette ligne, puis appuyez sur CTRL A jusqu'à ce que le curseur ait balayé toute la ligne. Faites RETURN, et c'est reparti !

En première position de PLAY doit figurer un chiffre de 0 à 7 commandant la "boîte de mixage" du synthétiseur selon les conventions suivantes :

0 synthétiseur bloqué
1 voie 1 en service pour la *tonalité*
2 voie 2 en service pour la *tonalité*
3 voies 1 et 2 en service pour la *tonalité*
4 voie 3 en service pour la *tonalité*
5 voies 3 et 1 en service pour la *tonalité*
6 voies 3 et 2 en service pour la *tonalité*
7 voies 3, 2 et 1 en service pour la *tonalité*

En seconde position, les mêmes codes revêtent la même signification, mais cette fois pour le *bruit de souffle.*

On peut donc affecter séparément des tonalités ou du bruit à chaque voie, voire mélanger les deux sur une seule voie.

La fréquence de la tonalité doit, bien sûr, être spécifiée au préalable par un ou plusieurs ordres SOUND ou MUSIC, faute de quoi seule la fréquence de 1 300 Hz pourra être obtenue.

En troisième position figure un code, compris entre 1 et 7, fixant la nature de l'enveloppe, à moins qu'un volume fixe n'ait été demandé dans une instruction SOUND ou MUSIC antérieure.

1 et 2 donnent des sons de percussion brefs. L'ordre PLAY doit être émis *chaque fois* qu'une percussion est souhaitée. Un ordre PLAY isolé et ne faisant pas partie d'une boucle ne donnera qu'un son unique.

Par contre, les codes 3, 4 et 6 donnent différentes variantes d'évolution "en dents de scie". Les sonorités ainsi obtenues sont permanentes et ne pourront être arrêtées que par un contre-ordre (tel que PLAY 0,0,0,0).

Le code 7 commande également des sons continus, mais affligés d'une attaque progressive (pas très utiles). Enfin, la quatrième position abrite un nombre compris entre 0 et 65535, déterminant la durée de variation de l'enveloppe (ou période), entre 1 ms et 17 s environ...

Un exemple d'utilisation d'un PLAY solitaire est fourni par le programme suivant :

```
10 INPUT V
20 PLAY 0,7,6,V
30 GOTO 10
```

La machine simule le bruit d'un petit train à vapeur, dont vous devez spécifier la vitesse en entrant des valeurs voisines de 500.

Voici par contre une application très complète des fonctions MUSIC et PLAY utilisées ensemble, comme il est normalement prévu de le faire :

```
10 MUSIC 1,3,1,0
20 MUSIC 2,3,5,0
30 MUSIC 3,3,8,0
40 PLAY 7,0,1,30000
```

Joli, n'est-ce pas ? En fait, l'ordre PLAY mixe ensemble les trois voies, sur lesquelles trois ordres MUSIC distincts envoient trois notes différentes, mais choisies de façon à créer un accord majeur.

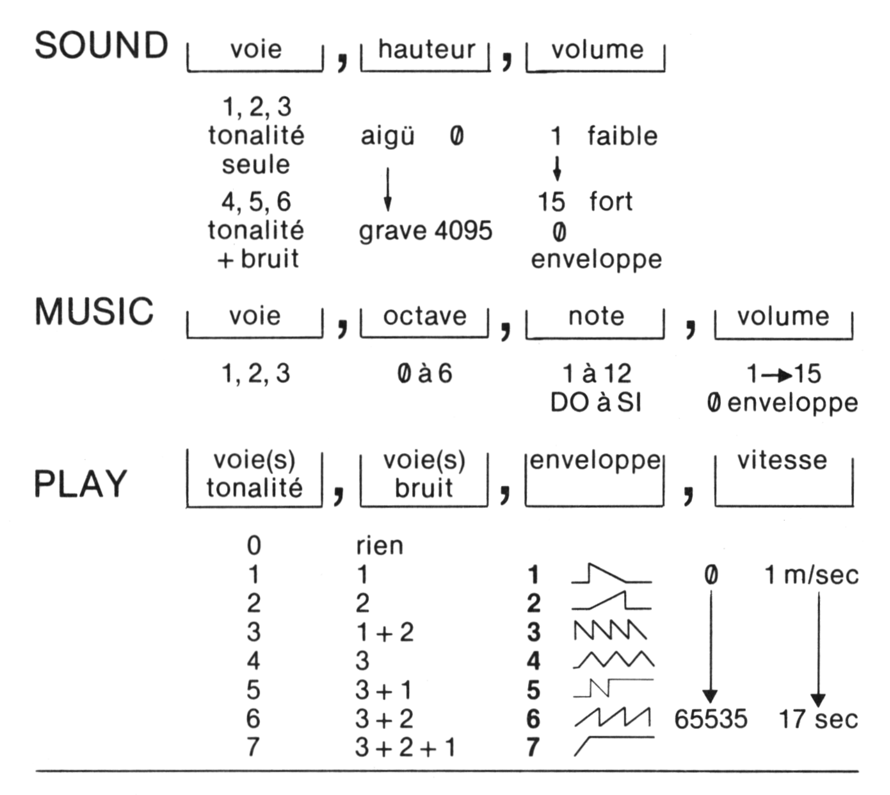


Figure 4-1 : *Les instructions sonores de l'ORIC.*

Le générateur de bruit est, bien sûr, neutralisé et une enveloppe décroissante très lente est demandée.

Sur le tableau de la *figure 4-1* sont regroupées les données essentielles régissant l'emploi des commandes SOUND, MUSIC et PLAY, dont les quelques exemples ci-dessus devraient avoir un peu démystifié l'usage !

Les raccordements

Le synthétiseur de sons de l'ORIC 1 attaque un haut-parleur incorporé qui ne peut hélas être déconnecté sans procéder à l'ouverture de la machine, donc à la rupture de la bande de garantie.

On peut cependant relier un amplificateur externe à la prise DIN à 7 broches prévue derrière l'ordinateur. Le brochage de cette prise *n'est pas standard,* ce qui nécessite quelques précautions :

Les deux broches du bas rejoignent le contact du relais de commande du magnétophone, et il n'y a donc pas lieu de s'en préoccuper. Les broches 1 et 3 correspondent respectivement à la sortie et à l'entrée du magnétophone, lequel peut donc être relié au moyen d'un cordon muni de fiches à trois broches. Un cordon 5 broches entraîne des phénomènes très désagréables dès lors qu'une opération CLOAD ou CSAVE est lancée.

On peut (et on doit !) cependant utiliser une fiche à cinq broches pour raccorder un ampli externe. En effet, ce sont les broches 4 et 5 qui reçoivent le son synthétisé ! Il faut donc soit câbler un cordon spécial reliant la broche 4 ou 5 de l'ORIC à l'entrée de l'ampli, soit utiliser un cordon standard à deux fiches cinq broches. Dans ce dernier cas, cependant, seul l'un des deux canaux stéréo fonctionnera. Le remède consiste tout simplement à actionner la commande MONO de l'ampli ! Rappelons-nous en effet que si le synthétiseur de l'ORIC dispose bien de trois canaux, il mixe ces trois signaux sur une sortie unique sans pouvoir créer un véritable signal stéréo. Même sans cette facilité, les effets obtenus sont souvent saisissants !

Si vous êtes un peu artiste

...Alors ce programme est fait pour vous !

```
10 GET A$
20 IF A$="1"  THEN N=1  : H=3
30 IF A$="2"  THEN N=2  : H=3
40 IF A$="3"  THEN N=3  : H=3
50 IF A$="4"  THEN N=4  : H=3
60 IF A$="5"  THEN N=5  : H=3
70 IF A$="6"  THEN N=6  : H=3
80 IF A$="7"  THEN N=7  : H=3
90 IF A$="8"  THEN N=8  : H=3
100 IF A$="9"  THEN N=9  : H=3
110 IF A$="0"  THEN N=10 : H=3
120 IF A$="-"  THEN N=11 : H=3
```

```
130 IF A$="="  THEN N=12 : H=3
140 IF A$="\"  THEN N=1  : H=4
150 MUSIC 1,H,N,0
160 PLAY 1,0,1,500
170 GOTO 10
```

Il transforme, en effet, les treize touches supérieures du clavier de l'ORIC en instrument de musique capable de jouer toute une octave de DO à DO, touches noires comprises.

Une sonorité de percussion brève a été choisie, mais il est très facile de la modifier en intervenant dans l'ordre PLAY selon les règles qui ont été développées plus avant.

Mais si vous ne l'êtes guère

...L'informatique peut tout de même quelque chose pour vous !

```
10 FOR N=1 TO 11
20 READ A,B
30 MUSIC 2,3,A,0
40 PLAY 3,0,1,2000
50 WAIT B
60 PLAY 0,0,0,0
80 NEXT
100 DATA 1,30,1,30,1,30,3,20,5,50,3,50
110 DATA 1,2,5,20,3,20,3,20,1,60
```

Cette fois, les notes à jouer ne sont plus frappées au clavier, mais sont incorporées dans le programme, grâce à de nouvelles instructions : READ et DATA.

Une instruction DATA est une liste de *données* que la machine vient lire à chaque exécution d'une instruction READ. Chaque READ fait progresser d'un cran l'exploration de la ligne DATA, à moins qu'une instruction RESTORE ne soit rencontrée, ce qui renvoie la lecture au début de DATA.

Dans notre exemple, l'instruction READ est double : READ A,B donnera ainsi la valeur 1 à A et la valeur 30 à B lors de sa première exécution. A la fin du programme, A vaudra 1 et B 60.

Il est intéressant de noter que les listes DATA ne sont pas limitées

en longueur, puisque plusieurs lignes peuvent s'enchaîner. Ici, par exemple, la ligne 110 est la suite pure et simple de la ligne 100.

Egalement, on peut "mélanger" dans une même liste de DATA, des données de nature différente, à condition de bien étudier le libellé des READ.

Ainsi, dans notre programme, vient-on lire hauteur de la note et durée du son à émettre dans la même ligne DATA.

Le petit air que joue ce programme ne comporte que onze notes, mais pourrait être considérablement plus long. Le remplissage de DATA à partir d'une partition ne pose pas de problème, car les notions de solfège nécessaires sont tout à fait élémentaires.

Pour des morceaux plus complexes, il faudrait, bien sûr, incorporer également des indications d'octave.

Chapitre 5

L'ORIC artiste peintre

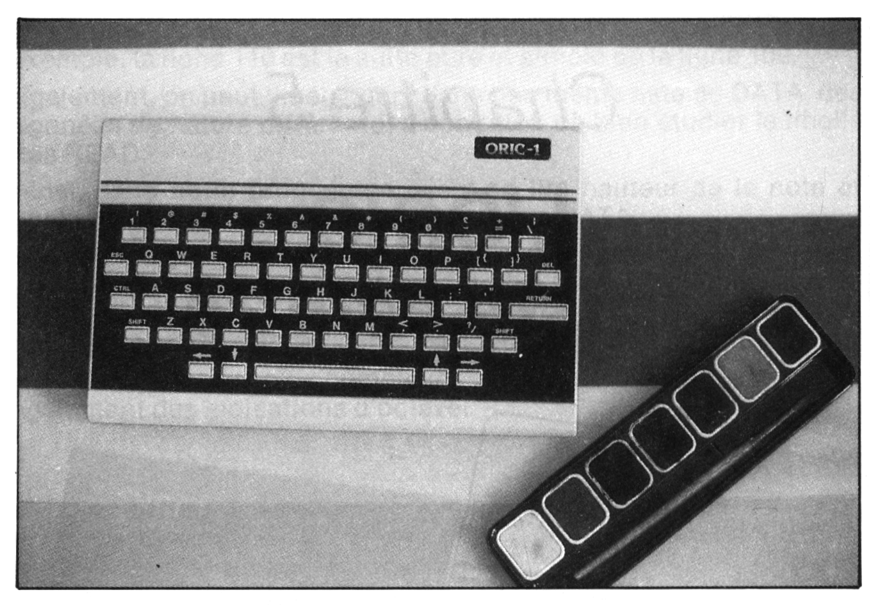


Parmi les multiples possibilités de l'ORIC figurent en bonne place de sérieuses aptitudes artistiques : nous avons déjà traité ici du synthétiseur sonore de cette machine, et notre propos sera maintenant de partir à la découverte de l'écran haute résolution couleur.

C'est bien d'une découverte qu'il s'agit, car le manuel reste très avare de précisions véritablement utilisables dans la pratique.

Les quatre modes d'affichage de l'ORIC

L'ORIC possède quatre modes d'affichage pouvant être appelés, soit par programmation, soit en mode commande, grâce à autant de mots-clé de son BASIC étendu : TEXT, LORES Ø, LORES 1, et HIRES.



A vos pinceaux, amis lecteurs !

Lors de sa mise sous tension, l'ORIC se trouve automatiquement commuté en mode TEXT, et se comporte presque comme une machine à écrire : à condition de ne pas frapper plus de 79 caractères à la suite, n'importe quel texte composé au clavier sera directement affiché à l'écran. Bien sûr, si RETURN est actionné, la machine répondra par une erreur de syntaxe si le texte frappé ne correspond pas strictement à une instruction BASIC.

A la mise sous tension, le clavier est "bloqué" sur les lettres CAPitaleS, comme en témoigne la mention CAPS en haut et à droite de l'écran.

Pour frapper un texte en minuscules (les majuscules pouvant toujours être appelées par SHIFT), il faut presser une fois et ensemble les touches CTRL et 7 (CTRL en premier). Si on répète la manœuvre, on reviendra en mode CAPS. Il s'agit d'une sécurité, car l'ORIC n'accepte pas les instructions BASIC libellées en minuscules !

Le mode LORES Ø permet de spécifier les coordonnées d'un emplacement sur l'écran (27 lignes de 39 caractères) auquel on imprimera un caractère, ou auquel on commencera à imprimer un groupe de caractères, au moyen de l'instruction PLOT :

```
10 LORES 0
20 PLOT 16, 12," ORIC
```

Remarquons, à l'occasion de cet exemple, qu'il est possible d'omettre le guillemet de fin de chaîne lorsqu'il est situé tout à fait en bout de ligne.

Le mode LORES 1 fonctionne de façon similaire, à ceci près qu'il est fait appel au *second jeu de caractères,* qui partage les mêmes codes ASCII avec le jeu standard.

Il s'agit des caractères graphiques normalisés du système de télétexte anglais VIEWDATA, ce qui permet aux utilisateurs britanniques de l'ORIC de se connecter, par téléphone, à de nombreuses bases de données grâce à un modem relativement bon marché.

"L'avance technologique" française en matière de télétexte se traduit par une procédure beaucoup plus complexe que celle utilisée outre-Manche, éliminant pratiquement tout espoir de pouvoir employer l'ORIC en tant que terminal télématique de façon raisonnablement simple...

Les deux jeux de caractères étant stockés en RAM, et donc entièrement redéfinissables sans l'aide d'aucun accessoire, il est facile d'accéder ainsi à une forme simplifiée de haute résolution graphique, très suffisante pour bien des applications dans le domaine des jeux : il sera très facile de déplacer sur l'écran un "petit monstre" construit à l'intérieur d'un caractère (ou de plusieurs caractères assemblés). Une animation pourra même être obtenue en faisant alterner plusieurs caractères légèrement différents (évolution de la position de membres, par exemple).

C'est cependant le mode HIRES qui nous intéresse plus spécialement ici, mais commençons par égayer un peu notre écran avec de la couleur !

La couleur sur l'ORIC

Il existe deux procédés permettant de colorier l'écran de l'ORIC, l'un très simple et l'autre très complexe. C'est bien sûr le second qui permet les effets les plus recherchés ! A la mise sous tension, l'ORIC délivre une image noir et blanc ("papier" blanc et "encre" noire). Les commandes INK et PAPER permettent très aisément de modifier les couleurs de l'encre, du papier, ou des deux, selon le code de la *figure 5-1.*

0 noir	**4** bleu
1 rouge	**5** magenta
2 vert	**6** cyan
3 jaune	**7** blanc

Figure 5-1 : *Code des couleurs INK et PAPER*

Essayons INK 7 (puis RETURN) : l'écran devient entièrement blanc quoi que nous tentions ! Rien de plus normal avec une encre blanche sur du papier blanc... Lançons un PAPER 0, et nous obtenons une sorte de "vidéo inversée" assez agréable. Un résultat assez sanglant peut être obtenu par INK 1, en conservant PAPER 0 : intéressant pour les "programmes d'horreur" !

Ce mode de coloration fort simple peut être utilisé sous les quatre modes d'affichage, mais n'autorise que deux couleurs simultanées alors que l'ORIC en possède huit (ou seize avec un regard d'agent de publicité !).

Pour spécifier une couleur autre que celle d'INK ou de PAPER en un point quelconque de l'écran, il faut utiliser la méthode des *attributs série.*

Un attribut est un code précédant un caractère, ou un groupe de caractères, pour en préciser la couleur, et certaines autres caractéristiques (clignotement, hauteur, etc.).

Seulement, ce code ne doit pas être reconnu et affiché comme un caractère normal !

La distinction entre caractères et attributs utilise un code spécial nommé ESCAPE, et qui possède d'ailleurs sa propre touche sur le clavier (CHR$(27)).

Tout simplement, un code précédé d'ESCAPE sera interprété non comme un caractère, mais comme un attribut, donc sans être affiché. Le procédé semble simple, mais sa mise en œuvre pratique est une toute autre affaire !

Il faut utiliser des tableaux analogues à celui de la *figure 5-2* pour le choix des attributs déterminant les couleurs INK et PAPER, et à celui de la *figure 5-3* pour ce qui concerne les autres caractéristiques des caractères dépendant de ces attributs. On peut utiliser un, deux, ou trois attributs consécutifs (chacun précédé d'un code ESCAPE, soit CHR$(27)), mais il faut savoir que le choix notifié en

Caractère après ESCAPE (27)	Code ASCII	Couleur PAPER (background)	Couleur INK (foreground)
@	**64**	↑	noir
A	**65**		rouge
B	**66**		vert
C	**67**	inchangée	jaune
D	**68**		bleu
E	**69**		magenta
F	**70**		cyan
G	**71**	↓	blanc
P	**80**	noir	↑
Q	**81**	rouge	
R	**82**	vert	
S	**83**	jaune	inchangée
T	**84**	bleu	
U	**85**	magenta	
V	**86**	cyan	
W	**87**	blanc	↓

Figure 5-2 : *Liste des attributs "couleur"*

Caractère après ESCAPE (27)	Code ASCII	Hauteur	Clignotant	Jeu de caractères
H	**72**	simple	non	normal
I	**73**	simple	non	graphique
J	**74**	double	non	normal
K	**75**	double	non	graphique
L	**76**	simple	oui	normal
M	**77**	simple	oui	graphique
N	**78**	double	oui	normal
O	**79**	double	oui	graphique

Figure 5-3 : *Liste des attributs "mode"*

début ou en cours d'une ligne-écran reste valable *jusqu'à la fin de cette ligne* ou jusqu'à un *contre-ordre* (de nouveaux attributs contraires aux précédents).

Seule l'étude de quelques exemples peut rendre raisonnablement claire l'utilisation des attributs :

Sans quitter le mode commande, frappons : [esc] RORIC.

Nous obtenons une ligne-écran uniformément verte, et le texte ORIC en noir. En effet, nous n'avons spécifié qu'un seul attribut : R, qui commande le vert de PAPER. Notons bien que, contrairement à un ordre PAPER qui affecte l'écran entier, l'attribut cesse d'agir à la fin de la ligne.

Bien sûr, si nous faisons RETURN, nous obtenons une erreur de syntaxe puisque ORIC n'a jamais été une instruction BASIC !

Pour imprimer un texte muni d'attributs en mode programme, il semblerait logique de faire :

10 PRINT " [esc] RORIC (puis RUN RETURN)

mais l'ORIC ne répond que par un RORIC noir sur blanc, car cette machine ne reconnaît pas la touche ESCAPE comme un caractère à part entière (il s'agit en fait d'une combinaison CTRL). Il faut donc faire appel au libellé suivant :

```
10 PRINT CHR$(27);"RORIC
ou
10 PRINT CHR$(27);CHR$(82); "ORIC
```

Le défaut n'est pas trop gênant avec un seul attribut en début de ligne, mais se fait plus que pesant avec trois attributs, ou même davantage lorsque l'on souhaite modifier un choix dans le courant d'une ligne : rappelons que, contrairement à ce qui se passe avec les machines SINCLAIR, il faut orthographier lettre à lettre C,H,R,$,(,2,7,),;! Bon courage...

A titre d'essai, amis lecteurs, convertissez donc en une ligne de programme la séquence suivante, qui fonctionne si bien en mode commande :

[esc] R [esc] A [esc] LORIC

(lettres rouges clignotantes sur un fond vert)

Pour limiter ce genre de corvée lors des manipulations qui vont suivre, nous avons écrit le court programme que voici, qui permet de faire "comme si" la touche ESCAPE fonctionnait comme il se doit :

```
10 A$=""
15 B$=KEY$
17 IF B$=CHR$(13) THEN 50
20 IF B$="" THEN 15
25 A$=A$+B$
40 GOTO 15
50 PRINT A$
60 GOTO 10
```

Il devient alors facile de frapper, après RUN, les combinaisons suivantes, séparées par RETURN. Attention, il n'est pas prévu de contrôle de la frappe sur l'écran, aussi le bruitage du clavier s'avérera fort utile.

[esc] R [esc] AORIC

procure un texte rouge fixe sur fond vert.

Essayons maintenant :

[esc] R [esc] A [esc] LVOICI [esc] WORIC

Le mot ORIC apparaît bien sur un fond blanc (qui se continue d'ailleurs jusqu'au bout de la ligne), mais ce texte est toujours rouge clignotant, car nous n'avons émis de contre-ordre que pour l'attribut PAPER.

De même :

[esc] R [esc] A [esc] LVOICI [esc] HORIC

perd seulement le clignotement dans sa seconde partie, les couleurs restant inchangées.

Rien n'empêche bien sûr de faire :

VOICI [esc] R [esc] A [esc] LORIC [esc] W

pour se contenter de mettre en valeur le seul mot ORIC. On remarquera alors, bien qu'aucun espace n'ait été frappé, un "blanc" de largeur équivalente à trois caractères entre les deux mots. Ils s'agit tout simplement des trois attributs R, A, et L qui, grâce à ESCAPE, ne sont pas affichés, mais sont bien là tout de même !

En début de ligne, on dispose d'une "franchise" de deux attributs, mais un troisième décalerait la ligne de texte d'un cran vers la droite.

Nous n'insisterons pas davantage sur l'utilisation de ces attributs, car ce n'est qu'au cours de multiples essais personnels que l'on finit par découvrir certains problèmes de *préséance* (l'ordre des attributs n'est pas toujours indifférent), et que l'on arrive à utiliser l'inénarrable fonction "double hauteur" : quelques bons moments en perspective, entre les lignes paires et impaires, la commande CTRL D et l'attribut CHR$(4), mais chut ! ne dévoilons pas le dénouement...

La haute résolution graphique

Dès que l'ordre HIRES est exécuté (par programme ou en mode commande), l'écran devient tout noir, à l'exception de trois lignes de texte en bas.

On peut, grâce à des instructions spécialisées, blanchir individuellement n'importe lequel des 240 × 200 points (ou "pixels") de l'écran haute résolution, ce qui permet d'obtenir des graphismes très détaillés.

Bien sûr, les commandes INK et PAPER permettent de choisir les deux couleurs qui seront mises en présence.

Dans presque toutes les instructions "HRG" (Haute Resolution Graphique), il sera fait usage d'un "code FB" (pour Foreground-Background). La *figure 5-4* donne la clé de ce code, légèrement différent selon le type de machine. En fait, le code FB sert uniquement à décider si l'opération à effectuer doit laisser une trace sur l'écran ou non, comme nous allons le découvrir :

Lançons le programme listé ci-après, qui n'utilise que la fonction CURSET (impression d'un pixel).

---

0 couleur du fond (PAPER)
1 couleur du devant (INK)
2 ATMOS : inversion de couleur INK et PAPER
ORIC 1 : couleur du devant (INK)
3 sans effet (aucun changement)

Figure 5-4 : *Code FB (Foreground - Background)*

```
5 HIRES : PAPER 4 : INK 3
10 X=(RND(1)*200)+20
20 Y=(RND(1)*160)+20
30 CURSET X,Y,1
50 GOTO 10
```

L'écran bleu va progressivement s'émailler de points jaunes placés aléatoirement.

Statistiquement, il suffirait d'attendre "suffisamment longtemps" pour obtenir un écran uniformément jaune.

Transformons la ligne 30 en :

30 CURSET X,Y,0

et l'écran demeurera imperturbablement bleu. Idée saugrenue ?

Pas du tout, car les points placés par CURSET servent souvent de référence à d'autres ordres, sans pour autant que l'on souhaite obligatoirement les voir apparaître à l'écran (centres de cercles par exemple).

Lançons ainsi le programme qui va suivre : de petits cercles (plus ronds sur l'ATMOS que sur l'ORIC 1...) vont progressivement couvrir tout l'écran.

```
5 HIRES : PAPER 3 : INK 4
10 X=(RND(1)*200)+20
20 Y=(RND(1)*160)+20
30 CURSET X,Y,0
40 CIRCLE 8,1
50 GOTO 10
```

Modifions la ligne 30 en :

30 CURSET X,Y,1

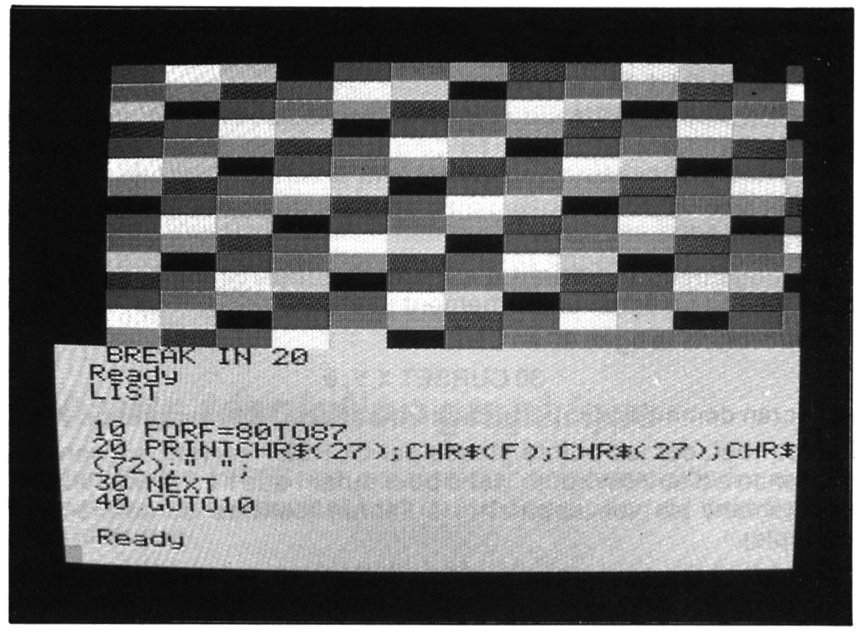
et les centres des cercles seront matérialisés, ce qui n'est pas forcément conforme au but recherché.

L'utilisation du même code FB dans une ligne supplémentaire :

45 CIRCLE R,0

permettra d'effacer les cercles sitôt tracés, ce qui peut constituer un effet intéressant.

Une petite variante permet aussi d'obtenir des cercles de rayons aléatoires, alors que le logiciel suivant rappelle certaines démonstrations souvent vues sur les salons !



Pour passer en couleur, lancez donc le programme présent à l'écran !

```
10 HIRES : PAPER 4 : INK 3
20 CURSET 110,90,1
30 FOR R=1 TO 60 STEP 2
40 CIRCLE R,1
50 NEXT
60 WAIT 200
70 RUN
```

La fonction CURMOV, pour sa part, permet de déplacer, suivant quatre axes, le point abandonné par CURSET. Si le code FB est bien choisi, une trace peut être laissée lors de tels déplacements, ce qui est exploité par ce programme, conçu pour mettre à profit les touches fléchées du bas du clavier. On pourra utiliser directement la fonction de répétition automatique pour tracer des lignes assez longues.

```
10 HIRES
20 CURSET 100,100,1
25 A$=KEY$
30 IF A$=CHR$(8 THEN CURMOV -1,0,1
40 IF A$=CHR$(9) THEN CURMOV  1,0,1
50 IF A$=CHR$(10)THEN CURMOV  0,1,1
60 IF A$=CHR$(11)THEN CURMOV  0,-1,1
70 GOTO 25
```

Bien sûr, cette ébauche de logiciel pourrait facilement être améliorée afin de permettre des déplacements en diagonale, ou des "levers de crayon".

Citons encore la fonction DRAW, capable de tracer des droites entre deux points quelconques de l'écran, presque aussi simplement que des cercles !

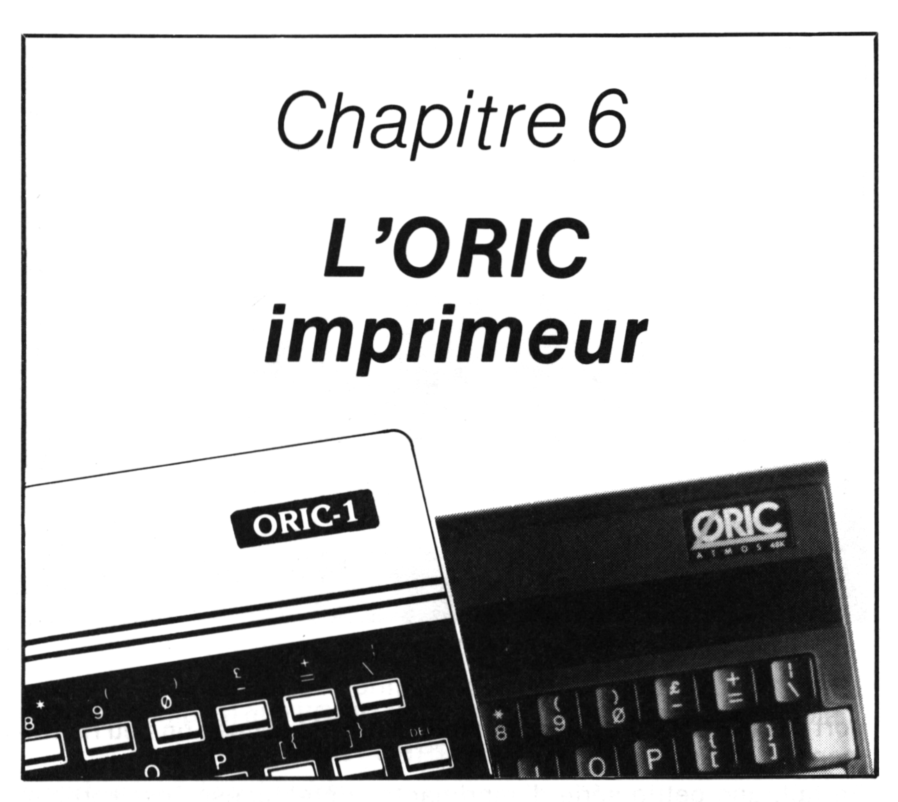
Conclusion

Voici, à notre sens, les bases qu'il est nécessaire de posséder pour tirer parti des ressources graphiques de l'ORIC. Nous n'avons bien évidemment pas épuisé le sujet qui est vaste et qui devrait apporter bien des satisfactions à nos lecteurs. Terminons avec le court programme reproduit ici, qui peut servir à une rapide mais impressionnante démonstration !

```
10 FOR F=80 TO 87
20 FOR G=64 TO 71
40 PRINTCHR$(27);CHR$(F);CHR$(27);CHR$(G);
50 PRINTCHR$(27);CHR$(76);"ORIC EN COULEUR"
60 NEXT G
70 NEXT F
```

Chapitre 6

L'ORIC imprimeur

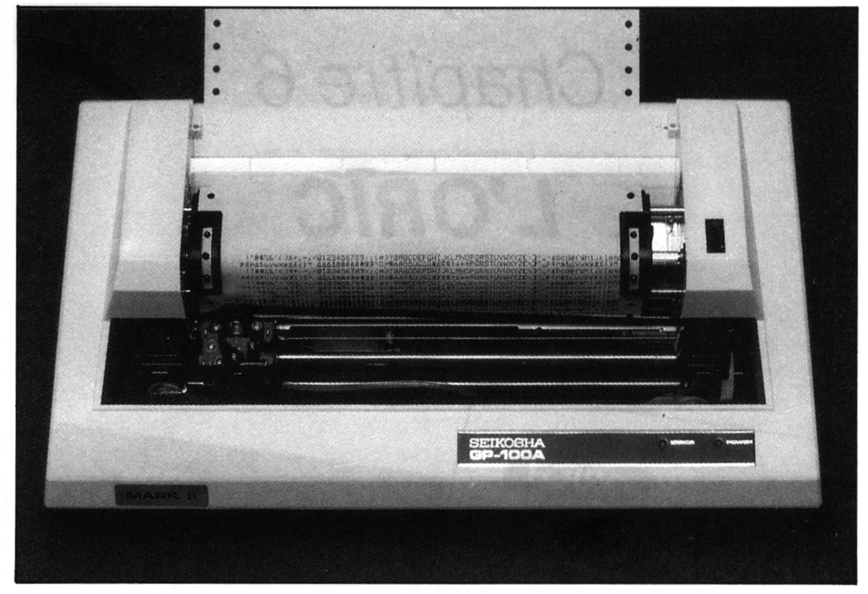


L'ORIC est l'un des ordinateurs individuels les plus répandus dans le grand public après le ZX 81, alors que l'imprimante SEIKOSHA GP 100 A semble s'imposer comme pratiquement la seule imprimante "papier ordinaire" d'un prix accessible à l'amateur.

Il est donc intéressant d'étudier les modalités d'association de ces deux machines, ainsi que les applications possibles du système semi-professionnel ainsi constitué.

Compatibilité entre l'ORIC et la GP 100 A

Un bruit étrange, venu d'ailleurs (de Grande-Bretagne), a couru quelque temps, selon lequel l'imprimante GP 100 A, lorsqu'elle était connectée à l'ORIC, bloquait complètement la machine ! Le



L'imprimante SEIKOSHA GP 100-A convient fort bien à l'ORIC.

port CENTRONICS de l'ordinateur a alors été accusé un peu hâtivement.

En fait, une petite série d'imprimantes défectueuses est sortie, à l'époque, des usines nippones, et le hasard a voulu qu'un confrère britannique en reçoive une, et le fasse savoir ! Aujourd'hui, l'incident est clos et nous avons pu vérifier le parfait fonctionnement de l'ensemble.

L'ORIC est muni d'une interface parallèle de type CENTRONICS qui, bien que simplifiée, permet le raccordement de toute imprimante conforme à ce standard très répandu.

Les instructions LPRINT et LLIST du BASIC permettent de faire parvenir à l'imprimante les caractères à imprimer.

Le manuel de l'ORIC 1 n'est guère bavard sur ce sujet, alors que celui de l'ATMOS ne jure que par l'imprimante ORIC (noblesse oblige...).

Une mise en garde est tout de même formulée, quant à la nécessité de vérifier la concordance des jeux de caractères des deux machines.

En effet, le manuel de l'imprimante montre que le jeu de caractères *imprimables* de la machine est pratiquement identique à celui de l'ORIC. Les rares exceptions (signe £ par exemple) ne sont guère gênantes pour un utilisateur français. On remarque, cependant, un jeu de caractères annexe (codes 160 à 222), et surtout des codes ASCII qui ne semblent correspondre à rien (en dessous de 32 et au dessus de 222). La plupart de ces codes sont, en effet, ignorés par l'imprimante, mais quelques-uns agissent en tant que *caractères de contrôle :* ils ne sont pas imprimés, mais commandent des fonctions spéciales de l'imprimante (retour à la ligne, avance papier, etc).

Avec l'ORIC, seuls deux de ces codes présentent un réel intérêt pour l'utilisateur :

CHR$ (14) sélectionne le mode "double largeur", fort utile pour mettre en valeur certaines portions d'un texte imprimé. CHR$ (15) ramène l'imprimante au mode normal "simple largeur". On notera bien que l'effet de ces commandes est définitif, jusqu'à réception du contre-ordre ou mise hors tension de l'imprimante pendant au moins deux secondes.

Ainsi, le lancement de la commande manuelle :

LPRINT CHR$ (14) RETURN

aura pour effet de faire imprimer en double largeur tous les textes (listings compris) que l'ORIC dirigera sur l'imprimante. Un simple :

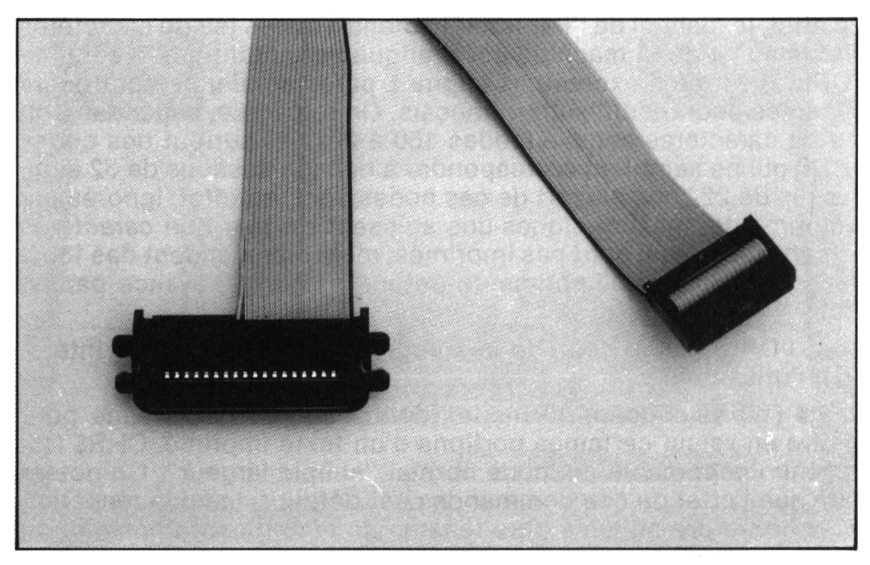
LPRINT CHR$ (15) RETURN

remettra tout dans l'ordre.

Bien entendu, ces fonctionnements supposent que l'ORIC 1 soit relié à la GP 100 A par un cordon d'interface approprié.

Raccordement de la GP 100 A à l'ORIC

L'imprimante SEIKOSHA GP 100 A est présentée par son importateur comme une machine pratiquement universelle, capable d'être "mariée" avec la plupart des ordinateurs du marché. Il existe, en effet, une large gamme d'accessoires spécialement étudiés pour assurer la liaison imprimante-ordinateur dans presque tous les cas de figure.



Un exemple de cordon du commerce (ELLIX informatique).

Le cas de l'ORIC est l'un des plus simples qui soient, puisqu'une interface CENTRONICS est incorporée dans la machine. Il suffit donc d'un simple cordon reliant « fil à fil » la prise normalisée de l'imprimante à celle, non normalisée, de l'ordinateur.

On trouve dans le commerce des cordons spécialement réalisés pour l'ORIC, à des prix variant du simple au double d'un point de vente à un autre.

Il est cependant très facile de fabriquer soi-même un tel câble, à condition d'arriver à se procurer les connecteurs nécessaires.

La *figure 6-1* fournit à cet effet les brochages des deux prises, les références des connecteurs à utiliser, et la liste des raccordements indispensables (11 fils).

Théoriquement, en effet, un seul fil de masse est suffisant, mais si la longueur du câble dépasse un mètre (deux au maximum), des phénomènes d'induction entre conducteurs voisins risquent de causer des perturbations.

Si du câble en nappe est utilisé, on conseille d'intercaler un fil de masse entre chaque conducteur « actif » et son voisin.

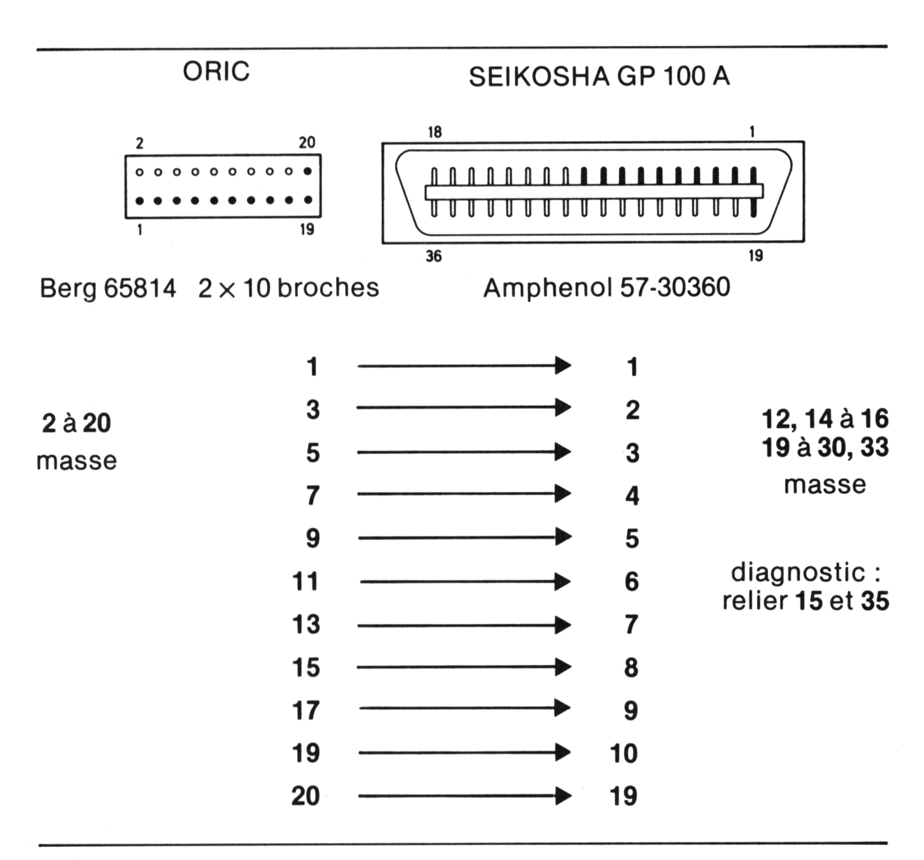


***Fig. 6-1 :** Connecteurs vus de face sur panneau arrière des machines*

Avec du câble multipaire (téléphonique), on n'utilisera pour les signaux qu'un seul fil par paire, le second étant relié à la masse.

C'est pour faciliter ce montage que de nombreuses broches de masse sont disponibles sur chacun des deux connecteurs.

Lors de la réalisation d'un tel cordon, il est bien sûr vital d'éviter toute erreur, dont les conséquences sur les machines pourraient être funestes...

En cas de doute sur la santé de l'imprimante, on pourra déclencher une procédure de test en reliant les broches 15 et 35 de son connecteur : une impression continue de tout le jeu de caractères démarrera alors, pour ne cesser que lorsque la liaison sera suppri-

mée. On peut ainsi localiser la panne éventuelle pouvant affecter l'imprimante, l'ordinateur, ou tout simplement le cordon.

Insistons sur le fait qu'une imprimante est un organe mécanique délicat, exigeant certains soins décrits dans la notice. On lira donc soigneusement ce document avant toute tentative d'utilisation !

Applications de l'ensemble ORIC-GP 100 A

L'adjonction d'une imprimante papier ordinaire à un ordinateur muni d'un clavier confortable ouvre la porte à toute une gamme d'applications semi-professionnelles, en plus du simple listage des programmes.

La qualité de la frappe de la GP 100 A, et la possibilité d'obtenir deux doubles carbone ou autocopiants permet d'utiliser le système en machine de traitement de texte ou en facturière simplifiée.

Ce n'est que pour le courrier "de prestige" que le recours à une machine offrant une plus belle frappe s'imposera vraiment. Egalement, le mode "double largeur" pourra s'avérer fort utile pour l'édition d'étiquettes d'expédition.

Le très simple logiciel qui va suivre transforme pratiquement l'ORIC en machine à écrire électronique : les marges peuvent être réglées à partir du clavier, même en cours de frappe grâce à la touche ESCAPE.

```
1 CLS:PRINTCHR$(20)
2 PRINT"LARGEUR D'IMPRESSION ? (caracteres)
";N
3 INPUTN
4 PRINT"MARGE DE GAUCHE ? (caracteres) ";M
5 INPUTM
6 B$="":FORF=0TOM
7 B$=B$+" "
8 NEXT
9 PRINT
15 GETA$
20 IFASC(A$)=13THEN100
25 IFASC(A$)=127THEN6
27 IFASC(A$)=27THEN2
```

```
30 PRINTA$;
40 B$=B$+A$
50 IFLEN(A$)>NTHENPING
55 IFLEN(B$)>66THEN100
60 GOTO15
100 CLS
110 PRINTB$
120 LPRINTB$
130 GOTO6
140 REM COPYRIGHT 1983 P.GUEULLE
```

Une sonnerie avertit l'opérateur qu'il atteint la limite qu'il s'est fixé, mais un retour à la ligne automatique intervient dès que la largeur imprimable risque d'être dépassée.

La ligne en cours de création est visualisée en permanence à l'écran, et ne sera imprimée sur papier que losque RETURN sera pressée, après contrôle visuel.

Si une erreur a été commise, la touche DEL permet de recommencer la ligne, sans pour autant que la version défectueuse ne se trouve effacée : une comparaison permanente est donc possible.

Généralités sur les systèmes de traitement de texte

Le principe fondamental sur lequel reposent tous les systèmes de traitement de textes consiste à séparer les fonctions SAISIE et IMPRESSION, autrefois réunies en une même opération appelée FRAPPE.

Lorsqu'un lien (mécanique ou électromécanique) rend solidaires les touches d'une machine à écrire et ses marteaux porte-caractères, la *frappe* d'une touche entraîne rigidement la *frappe* du caractère correspondant sur le papier.

Une erreur, un oubli, et tout le feuillet est à reprendre. Quelques modifications mineures à apporter à un texte souvent repris (lettres, actes, contrats, devis, rapports, etc), et il faut frapper à chaque fois la totalité du document.

En traitement de texte, les caractères "saisis" au clavier sont empilés dans une mémoire et simultanément visualisés sur un écran

vidéo. Des logiciels appropriés permettent très facilement d'intervenir sur le contenu de la mémoire, avec contrôle permanent sur l'écran.

Une fois le texte parfaitement au point, son transfert sur imprimante peut être demandé, et exécuté aussi vite que le périphérique le permet.

Bien des possibilités extrêmement appréciables peuvent se greffer sur ce canevas général :

Un texte peut être saisi par une secrétaire, relu sur écran et corrigé par une personne responsable, puis imprimé sitôt le "bon à tirer" donné. On élimine ainsi les énormes pertes de temps entraînées par les corrections sur des documents déjà frappés.

Bien plus, il est facile de conserver sur cassettes ou disques magnétiques, des modèles de textes "passe partout" dans lesquels on insèrera en peu d'instants les quelques lignes permettant de les *personnaliser.*

Pour peu qu'une imprimante "qualité courrier" soit utilisée (par exemple un mécanisme à marguerite ou à sphère), on obtiendra en un temps record une masse de circulaires dont l'aspect rappellera à s'y méprendre celui de la lettre personnelle.

Les notaires s'intéressent de très près à ces techniques, qui leur permettent d'éditer en un minimum de temps des actes bénéficiant d'une qualité de présentation sans égale.

Une variante du traitement de texte peut être trouvée dans toute imprimerie de quelque importance :

Depuis le remplacement de la typographie par l'offset, il n'est plus nécessaire d'aligner des caractères en plomb dans une forme. Des procédés de photogravure permettent d'exécuter l'impression directement à partir d'une maquette réalisée sur papier, en noir sur blanc.

Dès lors, les bruyantes linotypes ont cédé la place à des *photocomposeuses,* véritables ordinateurs qui ne diffèrent guère des machines de traitement de texte qu'au niveau de l'imprimante. En effet, les exigences de l'industrie graphique en matière de variété et de précision des caractères imposent le recours à des machines spéciales, fournissant le texte sous la forme d'un long ruban de papier photographique qui sera découpé à loisir par le maquettiste.

Ce texte est systématiquement *justifié,* c'est-à-dire traité de façon à aligner sur une même verticale les caractères de début et de fin de chaque ligne. C'est ainsi qu'est obtenue la présentation si agréable des colonnes de journaux et des pages de livres.

Bien sûr, les logiciels de traitement de texte cherchent à mettre à profit cette technique pour améliorer encore la qualité des documents élaborés.

Faut-il couper les mots ?

Il suffit d'examiner une page de livre ou de journal pour constater que la justification est obtenue, dans les photocomposeuses, au prix de la coupure de certains mots situés en fin de ligne. Or, on apprend en classe un certain nombre de règles régissant ces coupures, et qu'il ne saurait être question de transgresser sans produire un effet déplorable sur le lecteur.

Seulement, la programmation informatique de ces règles est extrêmement complexe, et ne saurait être introduite dans une machine de bureau dont le prix n'a rien de commun avec celui d'une photocomposeuse.

En traitement de texte, on se contente donc de revenir à la ligne à la fin d'un mot, en corrigeant le décalage existant nécessairement en fin de ligne par l'introduction d'espaces soigneusement répartis entre les différents mots de la ligne.

Bien sûr, cet artifice est d'autant plus visible que la ligne est courte, au point de devenir inutilisable dans le cas des étroites colonnes utilisées dans les journaux. En traitement de texte, cependant, on travaille le plus souvent sur toute la largeur d'un format A4, soit, marges déduites, quinze à vingt centimètres environ, et l'opération passe pour ainsi dire inaperçue.

La *figure 6-2* reproduit un échantillon de texte justifié à quarante caractères par ligne, sur un ensemble ORIC-GP 100 muni du logiciel qui va maintenant être décrit. Vis-à-vis de l'opérateur, la production d'un texte impeccablement justifié ne diffère en rien d'une opération habituelle de dactylographie, comme nous allons le voir.

Le logiciel de justification

Ce logiciel a été développé sur un ordinateur ORIC associé à une imprimante SEIKOSHA GP 100 A, et fonctionne en mode « machine à écrire électronique » : une ligne est imprimée dès que la longueur du texte saisi est suffisante.

```
Un   texte  imprime  gagne  beaucoup  en
qualite   de  presentation lorsqu"il est
justifie   a la maniere d"une colonne de
livre ou de journal.
La  justification consiste a aligner sur
une   meme  verticale, les caracteres de
debut et de fin de toutes les lignes.
On  obtient de cette facon des bandes de
texte parfaitement calibrees.
```

Figure 6-2

On dispose cependant de possibilités de correction tant que l'impression n'a pas eu lieu (correction ligne à ligne). Bien sûr, ce programme de base pourra facilement être incorporé dans un logiciel de traitement de texte offrant d'autres possibilités d'édition.

Le mode d'emploi, très simple, est le suivant :

Dès le lancement du programme, indiquer le nombre de caractères par ligne souhaité. On tiendra évidemment compte des limtations imposées par l'imprimante ou l'ordinateur (par exemple 67 caractères avec l'ORIC 1 et la GP 100 A).

Egalement, on évitera de justifier en dessous de 25 à 30 caractères par ligne, afin d'échapper à la mise en évidence de l'introduction d'espaces.

Quoi qu'il en soit, même si la longueur d'un mot venait à dépasser celle de la ligne, une coupure serait effectuée, mais au mépris de toutes les règles en la matière !

```
5 CLS:PRINTCHR$(20)
10 PRINT:PRINT:PRINT:PRINT
20 PRINT"LARGEUR D'IMPRESSION ?"
30 INPUT L
40 CLS
50 PRINT"FRAPPER LE TEXTE":PRINT:PRINT
55 B$=""
60 GETA$
```

```
62 IF A$=CHR$(13)THEN 600
65 IFA$=CHR$(127)THEN700
70 B$=B$+A$
80 PRINTA$;
90 W=0
100 IF LEN(B$)>L    THEN PING:PRINT;:GOTO 120
110 GOTO60
120 IF RIGHT$(B$,1)=CHR$(32) THEN LPRINT B$:
GOTO 55
130 Q=LEN(B$)
140 IF MID$(B$,Q,1)=CHR$(32)THEN200
150 Q=Q-1
160 IFQ=0THEN300
170 GOTO140
200 C$=LEFT$(B$,Q)
210 B$=RIGHT$(B$,LEN(B$)-Q)
220 Q=1
230 IFMID$(C$,Q,1)=CHR$(32)THEN C$=LEFT$(C$,
Q)+" "+RIGHT$(C$,LEN(C$)-Q):Q=Q+1
240 IFLEN(C$)=L THEN 400
250 Q=Q+1:IFQ<LEN(C$)THEN 230
255 W=W+1
257 IFW>100 THEN300
260 GOTO220
300 LPRINT LEFT$(B$,L-1);"-"
310 B$=RIGHT$(B$,2)
320 GOTO 60
400 IFRIGHT$(C$,1)=CHR$(32) THEN 500
410 LPRINT C$:GOTO 60
500 C$=LEFT$(C$,LEN(C$)-1)
510 GOTO 220
600 LPRINT B$
610 GOTO 55
700 CLS
710 B$=LET$(B$,LEN(B$)-1)
720 PRINTB$;
730 GOTO 60
1000 REM COPYRIGHT 1983 P.GUEULLE
```

Cela fait, le programme publié ici permet une saisie au clavier dans des conditions rappelant la dactylographie classique :

Les mots seront systématiquement séparés par des espaces, la touche RETURN ne devant servir que lorsqu'un retour à la ligne est souhaité.

Un signal sonore prévient du démarrage de l'impression : il faut alors cesser la frappe jusqu'à ce que la ligne soit transférée en entier sur le papier, mais il existe une marge de sécurité d'un caractère pour les virtuoses du clavier !

En cas d'erreur, la touche DEL permet d'effacer un caractère à la fois, ainsi que toute la partie de l'écran correspondant à du texte déjà imprimé, sur lequel il n'est évidemment plus question de revenir !

Il n'existe pas de limite à la longueur des textes pouvant être frappés grâce à ce logiciel, puisque les chaînes de caractères sont imprimées ou vidées bien avant que ne soit atteinte la limite fatidique (pour l'ORIC) de 256 caractères.

Conclusion

Ce logiciel est avant tout présenté en tant qu'illustration de la souplesse et de la commodité d'emploi de l'informatique en matière de dactylographie.

Il peut cependant trouver bien des applications pratiques pour peu que la machine sur laquelle il est chargé possède un clavier agréable à utiliser, et qu'une imprimante permettant une bonne qualité d'impression soit employée.

Il sera alors possible de donner une allure très proche de celle de textes photocomposés à bon nombre de documents.

Chapitre 7

L'ORIC robot ou à la découverte des interfaces



Les interfaces sont tout simplement les dispositifs annexes permettant à un ordinateur de communiquer avec autre chose que son clavier et l'écran TV.

Une *interface sonore* est incorporée d'origine dans l'ORIC, un *cordon d'interface* doit être utilisé pour connecter une imprimante, mais des accessoires appropriés peuvent établir des liaisons entre un ordinateur et une quasi-infinité de dispositifs externes, depuis les ampoules d'éclairage d'une pièce jusqu'à un poste téléphonique, en passant par un projecteur de diapositives !

Un fabricant français, déjà bien connu pour ses réalisations destinées au ZX-81, propose différents accessoires relativement peu coûteux permettant d'étendre presque à l'infini les possibilités de l'ORIC. Il s'agit de :

SIDENA
117, rue de la Croix-Nivert
75015 PARIS

Le module d'interface le plus riche d'applications est sans nul doute la carte à huit entrées et huit sorties "ORES", mais bien des satisfactions peuvent également être tirées de la carte analogique "OREA".

La robustesse et la fiabilité de l'ORIC ne sont nullement remises en cause par l'adjonction de ces modules, car leur raccordement s'effectue par un câble souple terminé par des connecteurs professionnels.

Ce cordon se connecte à la grande prise située près du connecteur destiné à l'imprimante, qui peut ainsi être branchée simultanément.

Prise de contact avec la carte ORES

La carte ORES est un circuit imprimé de qualité professionnelle équipé de trois robustes borniers à vis qui n'attendent qu'une chose : recevoir des connexions menant au milieu extérieur.

Huit petits voyants permettent cependant de travailler en "simulation" avant que les organes commandés ne soient branchés.

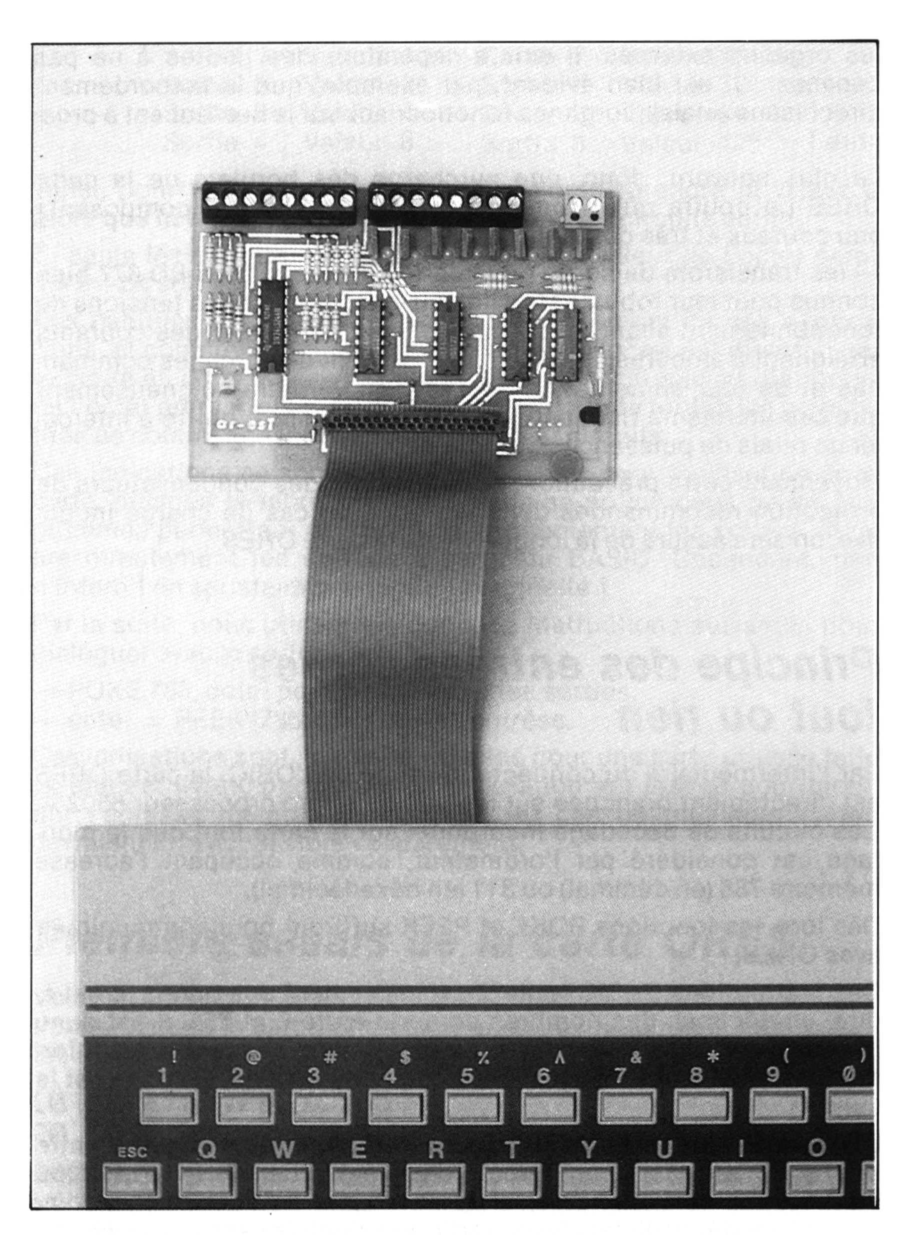
En effet, la carte ORES peut commander la mise en marche et l'arrêt de huit dispositifs indépendants : petits moteurs, électroaimants, et bien sûr relais autorisant le pilotage d'appareils alimentés sur le secteur.

En plus de cela, ORES peut interroger jusqu'à huit contacts rendant compte de l'état d'organes quelconques : boutons-poussoirs, fins de course, thermostats, détecteurs de niveau, barrières photoélectriques, etc.

Il est bien certain que la combinaison de ces possibilités d'entrée et de sortie ouvre la porte à des applications d'automatisation dont le nombre n'est guère limité que par l'imagination de l'utilisateur !

Avant que nous ne fournissions quelques exemples pratiques immédiatement réalisables, nous devons préciser deux points :

— la carte ORES protège normalement l'ORIC contre tout risque d'endommagement consécutif à de mauvaises manipulations sur



Une extension capable de décupler les possibilités de l'ORIC...

les organes externes. Il existe cependant des limites à ne pas dépasser : il est bien évident, par exemple, que le raccordement direct (sans relais) d'organes fonctionnant sur le secteur est à proscrire !

Le plus souvent, donc, une surcharge des borniers de la carte ORES ne pourra qu'endommager quelques-uns des composants peu coûteux et très courants qu'elle regroupe.

— les transistors de sortie de la carte ORES sont des BD 677 bien connus pour leur robustesse : ils peuvent accepter des tensions de fonctionnement allant jusqu'à 30 volts, et commuter des courants atteignant 2 ampères. Lors du raccordement des organes commandés et de leur alimentation séparée, on vérifiera soigneusement que ces limites ne risquent pas d'être dépassées, quitte à intercaler un relais de puissance appropriée.

Moyennant cette précaution, et l'utilisation des condensateurs de protection recommandés dans la notice en cas de charge inductive, on sera assuré de la longue vie de la carte ORES.

Principe des entrées-sorties tout ou rien

Par l'intermédiaire du connecteur arrière de l'ORIC, la carte ORES est directement branchée sur les bus du micro processeur 6502 A. Des circuits de décodage incorporés sur la carte font que le montage est considéré par l'ordinateur, comme occupant l'adresse mémoire 785 (en décimal) ou 311 (en hexadécimal).

Dès lors, les fonctions POKE et PEEK suffiront pour communiquer avec ORES.

Ces instructions BASIC traitent exclusivement des *octets,* c'est-à-dire, en décimal, des nombres compris entre 0 et 255. Il est donc nécessaire d'utiliser un *code* pour réunir dans ce nombre les informations décrivant l'état des huit circuits de la carte. Ce code est le suivant :

L'octet est obtenu en additionnant huit nombres, chacun d'entre eux étant affecté à un circuit d'entrée/sortie. Si la sortie (ou l'entrée) est inactive (à zéro), alors on affecte la valeur zéro au nombre qui en décrit l'état. Si la sortie (ou l'entrée) est active (à un), alors on affecte les valeurs suivantes au nombre correspondant :

Sortie 1	Valeur 1	Sortie 5	Valeur 16
Sortie 2	Valeur 2	Sortie 6	Valeur 32
Sortie 3	Valeur 4	Sortie 7	Valeur 64
Sortie 4	Valeur 8	Sortie 8	Valeur 128

Ainsi, pour tous les circuits inactifs, l'octet prendra la valeur 0, alors que la valeur 255 correspondra à tous les circuits actifs.

Si seuls les circuits 1,4, et 7 sont actifs, la valeur de l'octet sera : 1 + 8 + 64 soit 73.

Les habitués du binaire se passeront facilement de ces explications, et comprendront immédiatement que l'octet est directement exploité en code binaire pur par la carte. Les 256 valeurs distinctes de l'octet permettent bien de prendre en compte toutes les possibilités de commande indépendante des huit circuits.

Ces indications ne sont pas indispensables pour l'utilisation de la carte, puisque le fabricant donne dans sa notice deux petits programmes permettant de commander directement les sorties, et de lire directement les entrées à partir du BASIC. Cependant, rien n'interdit de satisfaire une légitime curiosité !

Par la suite, nous utiliserons donc les instructions suivantes pour dialoguer avec la carte :

— POKE 785, octet pour commander les sorties,
— octet = PEEK (785) pour lire les entrées.

Ces indications sont, bien sûr, valables pour une carte utilisée telle qu'elle est livrée, puisqu'une commutation est prévue pour modifier l'adresse mémoire de l'ORES, au cas où plusieurs cartes devraient être accessibles séparément.

Premiers essais de la carte ORES

Voici un très court programme permettant de vérifier le bon fonctionnement de la carte, et de faire quelques remarques intéressantes :

```
10 POKE785,PEEK(785)
20 GOTO 10
```

Sitôt le programme lancé, tous les voyants de la carte doivent s'allumer, et ses huit transistors de sortie doivent se saturer. Les huit sorties sont donc dans l'état « actif », qu'il est d'usage de noter "1" (charge alimentée).

Comme le programme se contente de retransmettre sur les sorties l'état des entrées, il est clair que toutes les entrées sont aussi à "1".

Cela est normal, puisque la carte utilise les niveaux logiques "TTL". Selon ces normes, une entrée "en l'air" est considérée comme étant à "1". Par suite, pour appliquer un "0" sur une entrée, il suffira de la relier à la masse, c'est-à-dire à l'une des vis du petit bornier à deux circuits.

Faisons subir ce traitement à chaque entrée de la carte, et observons l'extinction correspondante du voyant portant le même numéro (attention, les bornes ne sont pas dans l'ordre !) Tentons maintenant de commander les sorties d'ORES par les touches numériques de l'ORIC lui-même.

```
10 S=1 : GET A
15 IF A=9 THEN 10
16 IF A=0 THEN 10
20 FOR N=1 TO A
30 S=2*S
40 NEXT
50 POKE 785,S/2
60 GOTO 10
```

Si la lecture du clavier est une opération bien connue, si nous savons depuis peu imposer un état aux sorties d'ORES, nous ne sommes pas pour autant tirés d'affaire, puisque le manuel de l'ORIC ne mentionne nulle part l'existence de la fonction puissance !

Force nous est donc de la reconstituer au moyen de la boucle FOR-NEXT des lignes 20, 30 et 40, solution assez peu élégante, mais nous n'avions guère le choix ! (1)

C'est en effet au moyen d'une puissance de deux qu'il est possible d'obtenir l'octet correspondant au numéro de la touche pressée. Moyennant quoi, la machine donnera une bonne illusion d'un raccordement direct de ses touches 1 à 8 aux sorties de même rang de la carte ORES. La pression sur une nouvelle touche entraînera l'annulation de l'ordre précédent, ce qui ne permet donc que de valider une seule sortie à la fois.

Une approche un peu différente permet de commander séparément la marche et l'arrêt de chaque sortie :

---

(1) En fait, nous avons pu découvrir, beaucoup plus tard, que la puissance peut être obtenue (mais avec une lenteur consternante), au moyen du signe ∧ ...

```
10 DIM S(8)
20 GET A
30 GET A$
40 IF A$="M" THEN 100
50 IF A$="A" THEN 200
60 GOTO 30
100 S(A)=1
150 GOTO 300
200 S(A)=0
300 S=S(1)+2*S(2)+4*S(3)+8*S(4)
310 S=S+16*S(5)+32*S(6)+64*S(7)+128*S(8)
320 POKE 785,S
330 GOTO 20
```

Il suffit, avec ce programme, de presser la touche correspondant à la sortie sur laquelle on souhaite agir, puis sur M comme Marche ou A comme Arrêt selon le résultat désiré. Les lignes 300 et 310 contiennent une formule de conversion facilitant l'accès séparé aux huit sorties d'ORES, grâce à l'emploi de variables *indicées.*

Une variable indicée se compose d'une lettre suivie d'un nombre placé entre parenthèses et nommé indice. Par rapport à des variables telles que S1, S2 ou S3, S(1), S(2) ou S(3) présentent l'immense avantage que le nombre servant d'indice peut lui-même être une variable, ou même une expression qui sera évaluée le moment venu !

De plus, avec une seule lettre, on dispose d'un nombre à peu près illimité de variables.

Seule contrainte, il faut "déclarer" le nombre de ces variables avant de les utiliser, au moyen d'une instruction DIM. Ainsi, DIMS (8) réserve de la place en mémoire pour neuf variables notées S(0) à S(8), bien que nous n'utilisions pas S (0).

L'ORIC est aussi sourcilleux qu'un douanier et n'acceptera aucune variable indicée non déclarée. On se débarrasse donc généralement des formalités de déclaration dès les premières lignes d'un programme.

Notons qu'il ne s'agit là que d'une mise en œuvre très simple des *tableaux dimensionnés,* puisque plusieurs dimensions peuvent être spécifiées : DIM A (10, 10), par exemple, réserve de la place pour 121 variables.

On peut même dimensionner des chaînes de caractères : ainsi, DIM

A$ (20,5) déclare 126 caractères séparément identifiables par leurs indices.

Mais revenons à notre carte d'interface !

La formule des lignes 300 et 310 est capable de calculer l'octet à envoyer à ORES en fonction des valeurs 1 ou 0 affectées auparavant aux variables S(1) à S(8) accueillant les états des sorties 1 à 8. Quoi de plus pratique ?

Jusqu'à présent, nous nous sommes limités à des actions manuelles (par ORIC interposé tout de même) sur ORES. Il est maintenant temps de laisser plus d'initiative à la machine !

```
10 FOR F=0 TO 255
20 POKE 785,F
30 NEXT
```

Ce court programme "balaye" les 256 octets pouvant être envoyés à ORES. C'est dire que lors de son exécution, toutes les combinaisons possibles des états des huit sorties seront obtenues.

Ralenti par une ligne 15 ou 25 comportant un WAIT (aléatoire, pourquoi pas ?), ce programme pourrait commander les ampoules d'un jeu de lumière à huit canaux bien inhabituel !

Après les sorties, intéressons-nous un instant aux entrées :

```
5 CLS
10 DIM E(8)
20 A=PEEK (785)
30 K=128
40 FOR I=1 TO 8
50 E(9-I)=1
60 IF A<K THEN E(9-I)=0 : GOTO 80
70 A=A-K
80 K=K/2
90 NEXT
100 FOR F=8 TO 1 STEP-1
110 A$=A$+RIGHT$(STR$(E(F)),1)
120 NEXT
130 PLOT 9,10,A$
140 A$=""
150 GOTO 20
```

Ce programme affiche en permanence à l'écran les états des huit entrées d'ORES, permettant une surveillance permanente des capteurs qui lui sont reliés.

Bien sûr, rien n'empêcherait d'ajouter quelques lignes IF-THEN permettant à la machine de réagir dans certaines conditions bien particulières affectant l'état des entrées.

Un temporisateur de projection

Voici une première application pratique, bien que fort simple, de l'ensemble ORIC/ORES :

```
10 POKE 785,0
20 READ T
30 WAIT 100*T
40 POKE 785,1
50 WAIT 50
60 POKE 785,0
70 GOTO 20
80 DATA 5,2,10
```

Ce programme est destiné à commander le passage des vues sur un projecteur de diapositives, grâce à un relais branché sur la sortie n° 1 de la carte ORES.

Certains projecteurs permettent même un raccordement direct, subordonné à une sérieuse vérification des tensions, courants et polarités.

Il suffit de ranger dans la ligne DATA (ou dans plusieurs lignes DATA successives), la liste des durées de projection de chaque vue, exprimées en secondes.

Si le programme ainsi documenté est enregistré au début de la bande sonore du diaporama, il suffit de le charger en machine avant la projection pour que les opérations se déroulent de façon entièrement automatique, sans aucune synchronisation supplémentaire, comme nous l'avons démontré devant les caméras de TF1 en décembre 1983.

L'ORIC au téléphone

Les grands projets "télématiques" nous ont habitués depuis un moment déjà à l'association du téléphone et de l'informatique.

Si l'origine britannique de l'ORIC pose de sérieux problèmes de compatibilité avec les réseaux télématiques français, il ne faut

cependant pas désespérer d'arriver à connecter cette machine, d'une façon ou d'une autre, à une ligne téléphonique.

Auparavant, il faut bien entendu recueillir les autorisations administratives nécessaires et, si l'on ne se sent pas très familier avec les branchements téléphoniques, consulter notre ouvrage *Interphone, téléphone, montages périphériques,* paru chez le même éditeur.

```
5 POKE 785,0
10 READ C
15 IF C=0 THEN C=10
20 FOR N=1 TO C
30 POKE 785,1
40 WAIT 3.3
50 POKE 785,0
60 WAIT 6.6
70 NEXT
80 WAIT 50
90 GOTO 10
100 DATA 4,6,3,8,4,0,0
```

Ce programme de base permet à l'ORIC de composer automatiquement un numéro de téléphone dont les chiffres sont logés dans une ligne DATA.

Pour ce faire, il suffit d'alimenter un petit relais par la sortie n° 1 d'ORES, et de monter un contact *repos* de ce relais en série dans un des fils de ligne, ou mieux dans le circuit "I/II" du cadran du poste.

Ainsi, même lorsque le système ne sera pas en service, la ligne fonctionnera-t-elle tout à fait normalement.

La tonalité obtenue par un décrochage normal, il suffit de faire RUN pour que le numéro préprogrammé soit immédiatement composé.

Le principe de la numérotation est facile à comprendre sur cet exemple simple : la ligne, "bouclée" sur la résistance interne du poste, doit être ouverte et fermée selon un cycle normalisé de 33/66 millisecondes, un nombre de fois correspondant à chacun des chiffres du numéro composé (dix fois pour le zéro).

Il est facile d'étendre ce principe à des fonctions plus complètes :

```
5 POKE 785,0
10 GOTO 100
15 IF C=0 THEN C=10
20 FOR N=1 TO C
30 POKE 785,1
40 WAIT 3.3
50 POKE 785,0
60 WAIT 6.6
70 NEXT
80 WAIT 50
90 RETURN
95 STOP
100 A$="16/1 463 84 00"
110 FOR F=1 TO LEN (A$)
```

```
120 B$=MID$(A$,F,1)
130 IF B$=CHR$(47) THEN WAIT 500 : NEXT
140 IF B$=CHR$(32) THEN NEXT
150 C=VAL(B$)
160 GOSUB 15
170 NEXT
```

Ce programme permet l'introduction de pauses d'attente de tonalités intermédiaires (ligne extérieure, 16, 19), et autorise une présentation des numéros programmés plus agréable que dans une ligne DATA. En effet, il est possible d'insérer des espaces, qui seront ignorés à la lecture. Une pause de cinq secondes sera respectée chaque fois qu'un signe / sera rencontré. Des pauses plus longues seraient faciles à obtenir par accumulation de "/".

Une étape supplémentaire peut encore être franchie dans la direction d'un plus grand confort :

```
5 POKE 785,0
10 GOTO 200
15 IF C=0 THEN C=10
20 FOR N=1 TO C
30 POKE 785,1
40 WAIT 3.3
50 POKE 785,0
60 WAIT 6.6
70 NEXT
80 WAIT 50
90 RETURN
95 STOP
100 REM LECTURE CHAINE
110 FOR F=1 TO LEN (A$)
120 B$=MID$(A$,F,1)
130 IF B$=CHR$(47) THEN WAIT 500 : NEXT
140 IF B$=CHR$(32) THEN NEXT
150 C=VAL(B$)
160 GOSUB 15
170 NEXT
200 CLS : PRINT"NOM DE L'ABONNE ?"
210 INPUT N$
```

Suite au verso

```
220 IF N$="HORLOGE" THEN A$="463 84 00" :
GOTO 100
230 IF N$="POMPIERS" THEN A$="18" : GOTO 100
5000 PRINT "INCONNU AU BATAILLON !"
5010 PRINT "UN AUTRE NOM SVP"
5020 GOTO 210
```

Au lieu de frapper RUN pour composer l'unique numéro présent en mémoire, l'utilisateur pourra se limiter à indiquer à la machine le nom de son correspondant, et l'ORIC cherchera le numéro à composer dans un "annuaire" logé dans une suite de lignes IF-THEN.

En plus des lignes « échantillon » 220 et 230, l'utilisateur pourra entrer son propre répertoire.

Un correspondant est-il connu sous divers noms ou surnoms ? Il suffit, pour éviter de trop fréquents refus par la machine, de prévoir autant de lignes distinctes que de variantes possibles ! Même chose en ce qui concerne les fautes d'orthographe trop souvent commises !

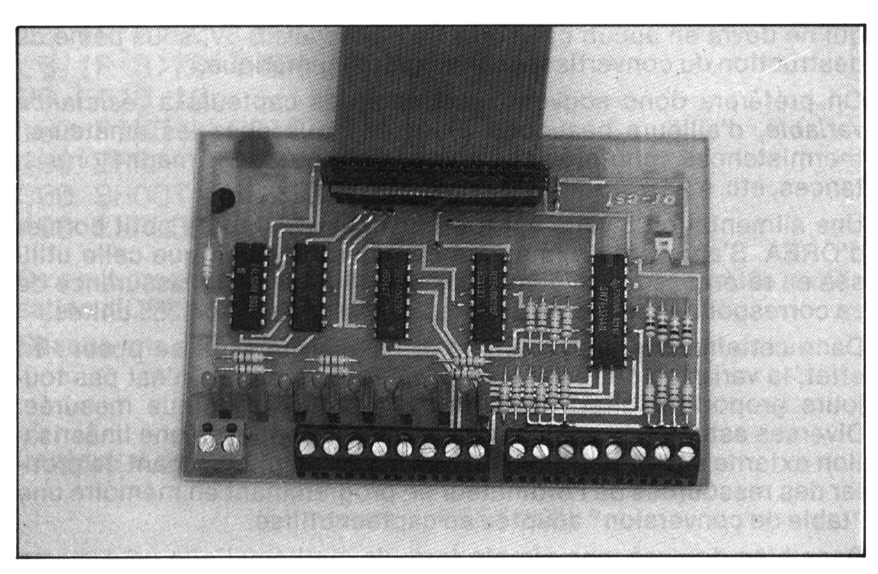
Notons dans ce programme l'utilisation de l'instruction REM. Une ligne commençant par ce mot peut accueillir n'importe quel texte, car elle sera ignorée à l'exécution. Cela est parfois précieux pour incorporer des REMarques de repérage dans le cours de longs programmes.

Prise de contact avec la carte OREA

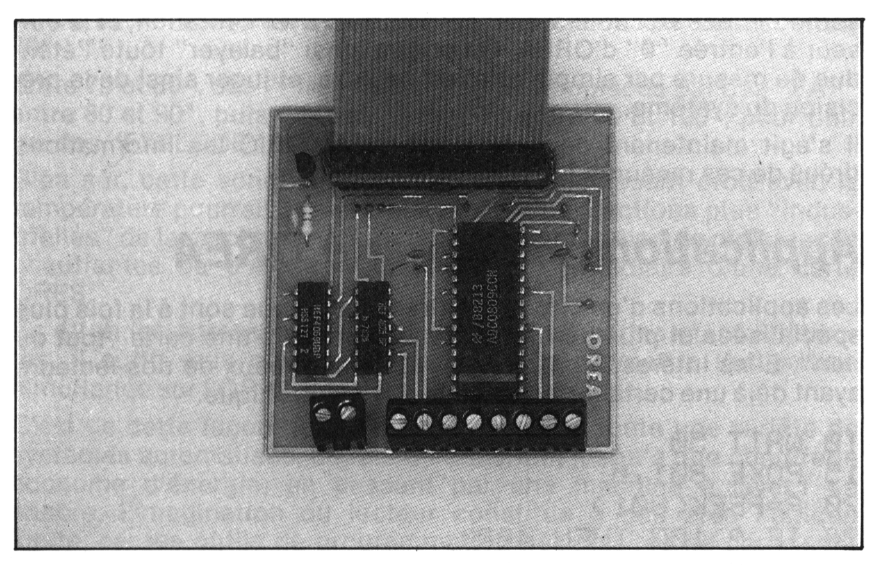
OREA est une carte nettement plus petite qu'ORES, et ne possédant que deux borniers. Elle ne comporte en effet que des entrées, toujours au nombre de huit, mais *analogiques.* Alors qu'ORES reçoit ses informations sous la forme d'ouverture ou de fermeture de contacts, OREA effectue de véritables mesures, transformant en nombres compris entre 0 et 255, des tensions évoluant entre 0 et + 5 volts.

La précision ainsi obtenue est meilleure que 0,5 %, ce qui peut être considéré comme très supérieur aux besoins de l'amateur : OREA pourrait fort bien prendre en charge des tâches de contrôle de processus industriels.

On pourra relier directement aux entrées d'OREA, des capteurs transformant n'importe quelle grandeur physique en une tension



La carte ORES munie de son câble plat.



La carte OREA munie de son câble plat.

qui ne devra en aucun cas sortir de l'intervalle 0-5V, sous peine de destruction du convertisseur analogique-numérique.

On préfèrera donc souvent recourir à des capteurs à *résistance variable,* d'ailleurs beaucoup plus répandus chez les amateurs : thermistances, photorésistances, potentiomètres, magnétorésistances, etc.

Une alimentation 5 volts est fournie à cet effet sur le petit bornier d'OREA. S'agissant de la même source de tension que celle utilisée en référence par le convertisseur, on peut avoir l'assurance de sa correspondance exacte avec la "pleine échelle" de 255 unités.

Dans certains cas, un problème de "linéarité" peut se poser : en effet, la variation de résistance des capteurs usuels n'est pas toujours proportionnelle à celle de la grandeur physique mesurée. Diverses astuces de branchement peuvent permettre une linéarisation externe acceptable, mais il est beaucoup plus élégant de profiter des ressources de l'ordinateur en programmant en mémoire une "table de conversion" adaptée au capteur utilisé.

Dans bien des cas, une simple formule mathématique peut même suffire, surtout si l'intervalle de mesure n'est pas très étendu.

Pour nos essais, nous pourrons remplacer le capteur par un simple potentiomètre de 4,7 kΩ A (linéaire), dont les extrémités de la piste seront reliées aux deux vis du petit bornier d'alimentation, et le curseur à l'entrée "0" d'OREA. On pourra ainsi "balayer" toute l'étendue de mesure par simple rotation de l'axe, et juger ainsi de la précision du système.

Il s'agit maintenant de faire exploiter à l'ORIC les informations tirées de ces mesures !

Applications de la carte OREA

Les applications d'un circuit d'entrée analogique sont à la fois plus spécialisées et plus performantes que celles d'une carte "tout ou rien". Elles intéresseront plus spécialement ceux de nos lecteurs ayant déjà une certaine expérience de l'électronique.

```
10 WAIT 50
15 POKE 801,0
20 A=PEEK(801)
30 IF A>100 THEN 400
40 IF A>90 THEN 300
```

```
50 IF A>80 THEN 200
60 IF A>70 THEN 100
70 GOTO 10
100 PING : GOTO 10
200 ZAP : GOTO 10
300 SHOOT : GOTO 10
400 EXPLODE : GOTO 10
```

Ce petit programme permet de mettre en évidence le mode d'utilisation d'OREA, à peine plus compliqué que celui d'ORES.

Chaque fois qu'une entrée doit être "interrogée", il faut faire un :

POKE 801, n° de l'entrée (entre 0 et 7)

Le résultat de la mesure sera ensuite lu par un :

mesure = PEEK (801)

Une fois ce résultat acquis, notre programme le compare à une série de "seuils", et déclenche une réaction de la machine adaptée à la situation.

On pourrait imaginer, par exemple, que l'ORIC surveille une température et ne reste silencieux que lorsque celle-ci ne dépasse pas 70°.

Entre 70 et 80°, des "PING" seront émis, remplacés par des "ZAP" entre 80 et 90°, puis par des "SHOOT" entre 90 et 100°, pour finir par des "EXPLODE" au dessus de 100°.

Bien sûr, cette sonorisation dont l'effet angoissant croît avec la température pourrait être remplacée par des réactions plus "industrielles" de la machine, par exemple la commutation de résistances chauffantes ou d'électro-vannes par l'intermédiaire d'une carte ORES.

En effet, les adresses des deux modèles de cartes étant différentes, il suffit d'un cordon spécial pour en permettre l'utilisation simultanée sur l'ORIC.

C'est de cette façon que l'on pourra réaliser toute une variété de systèmes automatisés, depuis un aquarium jusqu'à une chaufferie économe d'énergie, en passant par une machine à laver ! Là encore, l'imagination du lecteur constitue à peu près l'unique limite, car les outils de programmation nécessaires ne dépassent jamais le niveau de ceux fournis dans cet ouvrage.

Exploitation graphique des mesures

La combinaison du mode haute résolution de l'ORIC avec l'acquisition de résultats de mesures de grandeurs physiques rend possibles des visualisations dont la qualité n'a d'égale que la souplesse.

```
10 HIRES
15 FOR =0 TO 239 STEP 2
20 POKE 801,0
30 A=PEEK(801)
40 A=A/1.3
50 CURSET X,199,1
60 DRAW 0,-A,1
80 NEXT
90 GOTO 15
```

Ce programme construit un "histogramme" de l'évolution de la grandeur mesurée, mais sans l'effacer entre deux balayages de l'écran. On dispose donc en permanence, et même après des heures ou des jours de surveillance, d'un enregistrement des "crêtes", même brèves, qui se sont produites. Cette possibilité pourra se révéler intéressante dans le domaine météorologique (thermomètre à maxima), ou au niveau de la surveillance d'un congélateur.

Lorsqu'une plus grande clarté de lecture est souhaitée à tout instant (notamment dans le cas de phénomènes évoluant rapidement), on peut déclencher un effacement de l'écran dès que celui-ci est plein.

```
10 HIRES
15 FOR X=0 TO 239 STEP 2
20 POKE 801,0
30 A=PEEK(801)
40 A=A/1.3
50 CURSET X,199,1
60 DRAW 0,-A,1
70 WAIT 100
80 NEXT
90 RUN
```

En effet, en haute résolution, l'écran est vidé lors de chaque exécution de l'instruction HIRES (mais pas de CLS). Une instruction WAIT est prévue à la ligne 70 du programme, qui permet de régler la vitesse du balayage.

Comme l'argument de WAIT peut varier de 0 à 65535, cette ligne peut retarder jusqu'à dix minutes le passage d'une mesure à la suivante. Comme l'écran peut afficher 120 mesures (240 en supprimant STEP 2, pourtant utile au point de vue lisibilité), vingt à quarante heures peuvent être groupées sur un seul écran. Si cela ne suffit pas, rien n'empêche d'enchaîner plusieurs lignes WAIT...

Ces deux programmes exploitent l'instruction DRAW, dont le fonctionnement avait été passé sous silence dans le chapitre consacré à la haute résolution.

Cet ordre est exécuté par le 6502 en langage machine, ce qui est considérablement plus rapide qu'une boucle FOR-NEXT répétant un CURSET.

DRAW X, Y, 1 trace une ligne de la couleur de l'encre, entre le point spécifié par le CURSET précédent (ou toute autre instruction ayant déplacé le curseur haute résolution), et un point obtenu en ajoutant les *déplacements* X et Y aux coordonnées précédentes. Il s'agit donc d'une instruction *relative*.

Cependant, la présentation sous forme d'histogramme n'est pas nécessairement la mieux adaptée à tous les cas de figure.

```
40 HIRES
45 PRINT:PRINT"POUR ARRETER: CTRL C"
50 FOR X=0 TO 239 STEP 24
60 CURSET X,0,1
70 DRAW 0,199,1
80 NEXT X
90 FOR Y=0 TO 199 STEP 24
100 CURSET 0,Y,1
110 DRAW 239,0,1
120 NEXT Y
122 CURSET 239,0,1
128 DRAW 0,199,1
130 FOR X=0 TO 239
140 POKE 801,0
150 A=PEEK(801)
160 A=A/1.3
170 CURSET X,A,1
180 WAIT 10
190 NEXT
200 RUN
```

Voici une application très performante de l'informatique individuelle, permettant de reconstituer à très peu de frais le fonctionnement d'appareils coûtant plusieurs dizaines de milliers de francs.

Ce logiciel transforme, en effet, l'ORIC muni d'une carte OREA, en un oscilloscope à mémoire et à base de temps extrêmement lente. Sous réserve d'un étalonnage de l'écran gradué par rapport au capteur et à un chronomètre précis, on pourra disposer de représentations graphiques de très haute qualité, de l'évolution de phénomènes très lents.

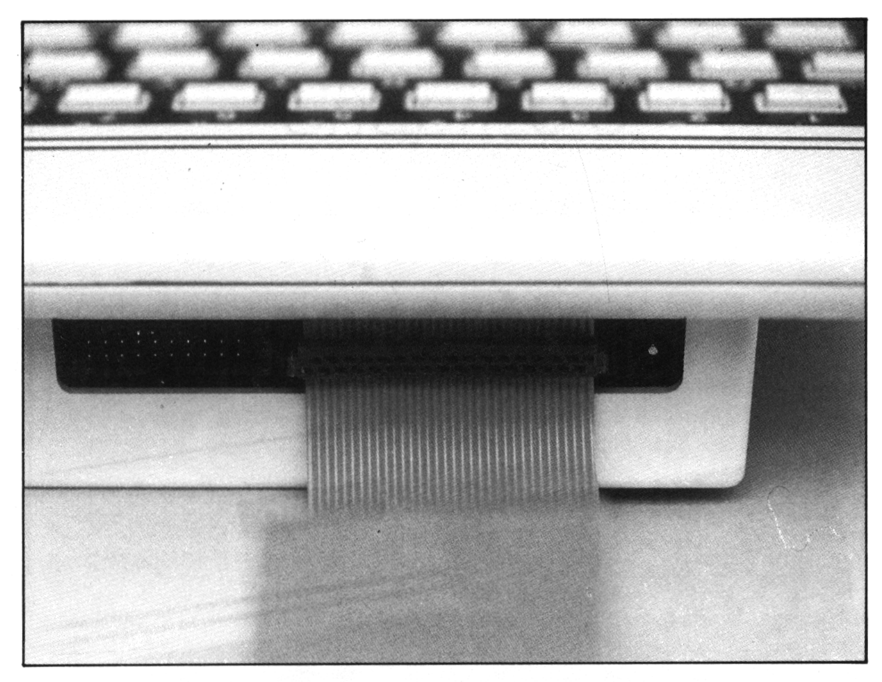
La ligne 180 permet de régler la base de temps, la ligne 160 la sensibilité verticale, alors qu'un simple CTRL C suffit pour figer l'image tout le temps nécessaire à son examen. Autrement, l'écran est effacé à la fin de chaque balayage, comme il est d'usage avec les oscilloscopes à mémoire. Le remplacement de la ligne 200 par GOTO 130 neutraliserait cet effacement, et permettrait un enregistrement des minima *et* maxima de la grandeur sur une durée aussi longue que nécessaire.

Les dimensions importantes d'un écran TV permettent une très grande finesse d'examen de l'image, difficile à obtenir même sur les meilleurs oscilloscopes.

On pourrait envisager très simplement l'introduction d'un curseur mobile (grâce aux touches fléchées), permettant d'aller lire une valeur remarquable de la courbe, mais... n'exagérons pas au point de quitter le cadre de cet ouvrage d'initiation !

Avec notre dernier programme, nous allons exploiter la carte OREA au maximum de ses possibilités, soit huit entrées.

```
40 HIRES
42 FOR V=0 TO 7
44 PRINT:PRINT "VOIE NUMERO ";V
50 FOR X=0 TO 239 STEP 24
60 CURSET X,0,1
70 DRAW 0,199,1
80 NEXT X
90 FOR Y=0 TO 199 STEP 24
100 CURSET 0,Y,1
110 DRAW 239,0,1
120 NEXT Y
122 CURSET 239,0,1
128 DRAW 0,199,1
130 FOR X=0 TO 239
140 POKE 801,V
150 A=PEEK(801)
160 A=A/1.3
170 CURSET X,A,1
190 NEXT
200 HIRES
210 NEXT V
220 RUN
```



Les cartes d'interface de l'ORIC se raccordent de façon particulièrement soignée et fiable.

Il s'agit d'une variante du programme précédent, permettant un examen alterné des huit grandeurs reçues par OREA. Le numéro de la voie affichée est rappelé dans la bande de texte disponible en bas de l'écran.

Bien sûr, la ligne WAIT a été supprimée, afin d'éviter un temps d'attente exagéré entre deux examens consécutifs de la même voie.

On pourra observer que même le bruit de fond, capté par les entrées laissées "en l'air", est visualisé sur l'écran. Ces expériences d'oscillographie informatique pourraient être poursuivies dans bien des axes : coloration de l'écran, visualisation simultanée de deux à huit traces, accélération massive de la base de temps grâce à une programmation en langage machine, et même recopie d'écrans sur imprimante, puisque la GP 100 A, par exemple, dispose d'un mode graphique. Relèverez-vous le défi, amis lecteurs ?

Chapitre 8

Quelques idées concrètes

Jusqu'à présent, les programmes de cet ouvrage ont été introduits de façon progressive, en synchronisme avec la découverte des principales instructions du BASIC de l'ORIC. C'est là, il nous semble, une façon attrayante d'apprendre les bases de l'informatique, grâce à la mise en œuvre pratique immédiate des notions nouvellement acquises.

Même si nous n'avons pas encore épuisé le "catalogue" des instructions de l'ORIC, nous disposons dès maintenant d'un nombre suffisant d'outils de programmation pour pouvoir envisager la concrétisation de bien des projets.

Bien évidemment, il ne nous appartient pas de nous substituer à l'imagination ou aux besoins de nos lecteurs ! Nous nous limiterons donc dans ce chapitre à présenter un certain nombre d'applications relativement inhabituelles de l'informatique individuelle, en nous efforçant d'obtenir des résultats honorables sans pour

autant faire appel à des subtilités de programmation qui risqueraient de dépasser le niveau atteint au fil des précédents chapitres. C'est dire que nos lecteurs ne devraient pas rencontrer de difficultés pour comprendre le fonctionnement profond de ces logiciels. Nous les encourageons d'ailleurs vivement à faire cet effort, car il leur permettra de mieux appréhender le comportement de l'ORIC dans les conditions les plus variées.
Cependant, les programmes qui vont suivre peuvent aussi être considérés comme des "produits prêts à l'emploi" : il suffit de les frapper au clavier pour pouvoir les utiliser immédiatement.
Dans ce cas, toutefois, une extrême vigilance est de rigueur lors de l'entrée en machine. Les occasions n'ont pas manqué, au fil des pages précédentes, de constater combien une modification insignifiante du libellé d'une instruction pouvait en perturber l'exécution : un espace en plus ou en moins, un point remplacé par une virgule, une parenthèse oubliée, et la machine peut se comporter de façon tout à fait imprévue.
Il est bien évident que les listes des programmes publiées ici ont été vérifiées et revérifiées. Un défaut d'impression est cependant toujours possible, rendant peu lisibles certains caractères. Dans de tels cas, heureusement fort rares, la confrontation de l'instruction douteuse avec les exemples antérieurs doit permettre de lever l'ambiguïté.

L'ORIC et les messages secrets

Codage et décodage de messages "secrets" ont toujours été une source inépuisable de divertissements pour les jeunes et... les moins jeunes.

En milieu militaire, administratif ou industriel, la *cryptographie* ou science du codage de l'information, est très largement utilisée.

Les techniques cryptographiques n'ont pu se perfectionner massivement que grâce à l'introduction de la "grosse informatique" dans les "services du chiffre".

L'emploi d'ordinateurs décharge le personnel des manipulations extrêmement fastidieuses de codage et de décodage, mais également la programmation en machine de règles statistiques fort complexes offre des moyens puissants pour tenter de décrypter des messages dont la clef n'est pas connue ! Sans mettre en œuvre des principes par trop délicats, nous allons décrire ici quelques méthodes facilement applicables sur un ordinateur individuel tel que l'ORIC.

• Les codages par substitution

Tous les scouts ont un jour utilisé un code par substitution analogue à ceux en usage pendant la première guerre mondiale : chaque lettre de l'alphabet est remplacée par une autre, toujours la même.

C'est le "tableau de correspondance" qui constitue la clef du code.

Cependant, l'expérience montre qu'il n'est pas indispensable de coder toutes les lettres, mais seulement les plus fréquemment utilisées, notamment les voyelles.

Pour s'en convaincre, il suffit de lancer le programme suivant :

```
10 C$=KEY$ : D$=C$
20 IF C$="A" THEN D$="R"
30 IF C$="E" THEN D$="N"
40 IF C$="I" THEN D$="L"
50 IF C$="O" THEN D$="U"
60 IF C$="U" THEN D$="O"
70 IF C$="L" THEN D$="I"
80 IF C$="N" THEN D$="E"
90 IF C$="R" THEN D$="A"
100 PRINT D$;
110 GOTO 10
```

et de frapper un texte quelconque, qui apparaîtra codé sur l'écran, comme le montre cet exemple :

```
CET ORDINATEUR EST UN ORIC

CNT URDLERTNOA NST OE UALC
```

Comme le cryptage est fait lettre par lettre avec affichage immédiat, il n'existe aucune limite quant à la longueur des textes pouvant être traités.

A partir d'une certaine importance, cependant, il devient très pénible de recopier à la main et sans erreur un texte codé, qui est par définition à peu près illisible.

Egalement, l'absence de contrôle visuel de la frappe peut poser des probèmes.

Les possesseurs d'une imprimante pourront avantageusement préférer la variante suivante :

```
10 C$=KEY$ : D$=C$
20 IF C$="A" THEN D$="R"
30 IF C$="E" THEN D$="N"
40 IF C$="I" THEN D$="L"
50 IF C$="O" THEN D$="U"
60 IF C$="U" THEN D$="O"
70 IF C$="L" THEN D$="I"
80 IF C$="N" THEN D$="E"
90 IF C$="R" THEN D$="A"
100 PRINT C$;
110 LPRINT D$;
120 GOTO 10
```

Chaque caractère frappé sera affiché tel quel sur l'écran, alors que le texte codé sera transmis à l'imprimante.

En raison de la présence d'un "registre tampon" dans la majorité des imprimantes (c'est le cas pour la GP 100), une ligne n'est imprimée que lorsqu'elle est complète.

On veillera donc à bien presser RETURN en fin de texte pour que la dernière ligne, même incomplète, soit imprimée.

L'un des avantages de ce très simple logiciel de codage est son fonctionnement symétrique : A devient R, mais R devient A. De ce fait, il peut servir sans la moindre transformation à *décoder* les messages qu'il a lui-même élaborés. Il suffit de frapper le texte codé pour que la traduction s'affiche en clair sur l'écran TV ou sur l'imprimante, selon la variante utilisée.

Le tableau de transcodage des lignes 20 à 90 pourra évidemment être transformé par chacun de façon à créer une clef originale. On veillera simplement à ne pas détruire la symétrie du code, faute de quoi un programme spécifique serait à écrire pour le décodage.

• Peut-on décrypter sans la clef ?

Posée à un spécialiste des services secrets, cette question recevrait sans doute la réponse suivante : "oui, mais à condition de disposer de beaucoup de temps et de moyens".

En effet, les services de contre-espionnage du monde entier opèrent chaque jour ce genre de prouesse, grâce à de puissants ordinateurs et à des programmes ultra-sophistiqués. En ce qui concerne l'amateur, des résultats encourageants peuvent être obtenus dans des conditions bien précises :

```
10 DIM T$(8000)
20 DIM C(125)
30 DIM D(10)
100 GET A$
110 A=ASC(A$)
115 IF A=13 THEN 200
120 C(A)=C(A)+1
130 PRINT A$;
132 T$(Q)=A$
134 Q=Q+1
140 GOTO 100
200 CLS
205 FOR I=1 TO 10
210 FOR F=65 TO 90
220 IF C(F)>M THEN M=C(F) : D(I)=F
240 NEXT F
242 C(D(I))=0
245 M=0
250 NEXT I
260 FOR Q=0 TO 8000
270 Y$=T$(Q)
280 IF Y$=CHR$(D(1)) THEN T$(Q)="E"
290 IF Y$=CHR$(D(2)) THEN T$(Q)="S"
300 IF Y$=CHR$(D(3)) THEN T$(Q)="A"
310 IF Y$=CHR$(D(4)) THEN T$(Q)="R"
320 IF Y$=CHR$(D(5)) THEN T$(Q)="I"
330 IF Y$=CHR$(D(6)) THEN T$(Q)="N"
340 IF Y$=CHR$(D(7)) THEN T$(Q)="T"
350 IF Y$=CHR$(D(8)) THEN T$(Q)="U"
360 IF Y$=CHR$(D(9)) THEN T$(Q)="L"
370 IF Y$=CHR$(D(10))THEN T$(Q)="O"
380 PRINT T$(Q);
390 NEXT Q
400 REM COPYRIGHT 1983 P.GUEULLE
```

Ce logiciel expérimental procurera certainement bien des satisfactions à l'amateur curieux mais réaliste.

Il lui faudra en effet comprendre qu'un programme aussi simplifié ne peut en aucun cas opérer de miracles, mais seulement décryp-

ter (sans la clef !) des messages écrits au moyen d'un code de substitution analogue à celui que nous avons publié, et sous certaines réserves. Moyennant quoi, de saisissantes démonstrations peuvent être réalisées...

Le principe de ce logiciel est le suivant :

Dans toutes les langues courantes, certaines lettres sont employées plus souvent que d'autres.

L'analyse statistique d'importants volumes de texte permet même de classer par ordre décroissant de fréquence d'apparition les dix lettres les plus utilisées.

En français, cet ordre est E, S, A, R, I, N, T, U, L, O. Dès lors, lorsqu'un cryptage *par substitution simple* est mis en œuvre, il est théoriquement possible de comptabiliser tous les caractères utilisés, puis de remplacer celui le plus souvent rencontré par E, le suivant par S, etc., pour se rapprocher très fortement du texte décrypté.

L'inconvénient de la méthode est qu'elle ne fonctionne correctement que sur des textes de longueur notable, car autrement les règles statistiques (loi des grands nombres !) ne s'appliquent que très imparfaitement.

Egalement, la présence dans le texte original, de mots techniques, ou empruntés à des langues étrangères, perturbe le décodage.

Nos lecteurs pourront expérimenter dans ce passionnant domaine en codant des textes d'origines et de longueurs diverses au moyen du logiciel précédemment présenté, puis tenter de les décrypter à l'aide de ce nouveau programme. Si le codage est effectué par une personne et le décodage par une autre, le jeu risque de devenir fort distrayant, surtout lorsqu'un travail supplémentaire d'interprétation doit être accompli par le décrypteur (cas de messages trop courts).

L'utilisation du logiciel de décodage sans clef est aussi simple que celle du précédent : il suffit de frapper le texte à décrypter, avec contrôle visuel à l'écran, et de presser RETURN à la fin. Le résultat du décodage s'inscrit alors au fur et à mesure de l'avancement des opérations.

Bien sûr, une modification simple pourrait permettre de sortir ce résultat sur imprimante, ou même de traiter des textes comportant d'autres signes ASCII que des lettres (il faudrait alors transformer la ligne 210).

On évitera de frapper le texte à décoder à une vitesse exagérée, car des calculs doivent être faits entre deux caractères successifs. Pour éviter toute surprise, on attendra simplement, pour frapper un caractère, que le précédent soit apparu à l'écran, ce qui n'est pas bien long !

• Un codage plus subtil

Nous avons pu constater comment même des moyens informatiques réduits arrivent à percer les codes par trop naïfs.

Voici un procédé de cryptage tout différent, qui résistera victorieusement à notre logiciel de décodage.

Son principe est issu d'une technique très simple, dite de la feuille quadrillée :

Le texte à coder est écrit, case par case, dans une grille carrée. Les éventuelles cases inutilisées sont remplies avec des caractères quelconques, puis la transcription est effectuée en "lisant" la grille non plus selon des lignes horizontales mais selon des colonnes verticales.

Le brassage ainsi exécuté revient pratiquement à battre un jeu de cartes, mais pour que l'opération inverse ramène aisément au texte d'origine, encore faut-il connaître le nombre de cases de la grille, véritable clé du code.

Rien n'empêche d'ailleurs de scinder le texte en plusieurs parties, qui seront chacune codées selon leur propre grille, et même d'opérer auparavant un codage par substitution... C'est exactement selon cette méthode qu'opère le programme que voici :

```
1 CLS
2 PRINT "CLE DU CODE ?"
3 INPUT K
5 CLS : PRINT "ATTENDEZ..."
10 DIM T$(K,K)
20 FOR L=1 TO K
30 FOR C=1 TO K
40 N=65+(25*RND(1))
50 T$(L,C)=CHR$(N)
60 NEXT C
70 NEXT L
72 CLS
```

Suite au verso

```
75 PRINT "FRAPPEZ VOTRE MESSAGE"
77 PRINT
80 FOR L=1 TO K
90 FOR C=1 TO K
92 GET A$
94 IF A$=CHR$(13) THEN 125
96 T$(L,C)=A$
105 PRINT A$;
110 NEXT C
120 NEXT L
125 CLS : PRINT
130 FOR L=1 TO K
140 FOR C=1 TO K
150 PRINT T$(C,L);
160 NEXT C
170 NEXT L
180 REM COPYRIGHT 1983 P.GUEULLE
```

La “clé du code” (nombre de cases du côté de la grille) doit être choisie en rapport avec l’importance du texte à coder : élevée au carré, elle doit donner un résultat supérieur au nombre de caractères du texte à coder.

La grille est d’abord entièrement remplie de caractères choisis aléatoirement, ce qui prend un certain temps (message “attendez...”). Le message frappé au clavier vient “écraser” ces caractères jusqu’à concurrence de sa longueur, mais pas au-delà.

La grille est ensuite relue en permutant lignes et colonnes, et le texte codé ainsi obtenu est affiché à l’écran. Une sortie sur imprimante serait bien sûr facile à prévoir. Le texte crypté est élaboré dès que la touche RETURN est actionnée en fin de texte, ou bien en cas de débordement accidentel de la grille.

Cette grille n’est autre qu’un tableau multidimensionnel de caractères, réservé par une instruction DIM.

Voici un exemple des résultats obtenus :

```
MGC IEERHIS EHASSTEXAE YU


MESSAGE SECRET  HHEYIIAXU
```

L'ORIC duplicateur

Dans les exemples précédents, nous avons vu que l'ORIC n'accepte que des chaînes de caractères de longueur très limitée.

Pour s'affranchir de cette difficulté, certains de nos précédents programmes ont fait appel au traitement individuel et immédiat des caractères frappés, d'autres se sont contentés de traiter du texte ligne par ligne, alors que les derniers présentés ont fait appel à des tableaux dimensionnés, dont le nombre d'éléments n'est guère limité que par la capacité mémoire de la machine.

Le programme qui va être présenté ici exploite lui aussi cette dernière solution, pour mémoriser toute une page de texte, qui pourra ainsi être rappelée à loisir.

• Applications d'un programme de duplication

Un logiciel de duplication permet à un ordinateur équipé d'une imprimante, d'éditer un nombre illimité d'exemplaires "originaux" d'un texte préalablement saisi au clavier.

Si l'imprimante utilisée possède une frappe de "qualité courrier" (à marguerite ou à sphère), chaque exemplaire passera aisément pour une lettre personnelle, ce qui ne serait pas le cas pour une photocopie.

Mieux encore, en greffant sur le programme des routines *d'insertion,* la machine pourra "personnaliser" chaque copie à partir, par exemple, d'un fichier d'adresses.

Même avec une imprimante à "matrice de points", l'aspect d'un premier et unique exemplaire est toujours nettement meilleur que celui d'une quelconque copie.

Nous nous limiterons ici à l'étude du procédé permettant de mémoriser un texte saisi au clavier, puis de le sortir en entier sur imprimante, autant de fois que nécessaire. Les éventuelles adjonctions de facilités supplémentaires telles que justification et insertion ne posent guère de problèmes, mais dépendent étroitement de l'application envisagée.

- **Mise en œuvre pratique**

```
5 CLS : PRINT CHR$(20);"FRAPPER LE TEXTE" :
PRINT
10 DIM A$(66,50)
20 FOR L=0 TO 50
30 FOR C=0 TO 66
35 GET B$
40 IF B$=CHR$(13) THEN 100
41 IF B$=CHR$(127) THEN CLS : GOTO 30
42 IF B$=CHR$(27)THEN PRINTCHR$(20):GOTO 200
45 PRINTB$; :A$(C,L)=B$
50 NEXT C
60 PING
65 PRINT
70 NEXT L
100 A$(C,L)=CHR$(10)
110 C=66
120 GOTO50
200 CLS
205 FOR L=0 TO 50
210 FOR C=0 TO 66
220 LPRINT A$(C,L);
230 NEXT C
240 NEXT L
250 CLS : PRINT:PRINT"ENCORE
UNE IMPRESSION ? O/N"
260 GETC$
270 IFC$="O" THEN 200
280 IFC$="N" THEN RUN
290 GOTO 260
300 REM COPYRIGHT 1983 P.GUEULLE
```

Le logiciel listé ici a été étudié spécialement pour fonctionnner sur un ensemble ORIC/GP 100 A Seikosha. Il a donc été prévu pour accepter jusqu'à 50 lignes de 66 caractères, sauf retour à la ligne anticipé.

Il est bien évident que les lignes 10, 20, 30, 110, 205 et 210 pourraient facilement recevoir d'autres "réglages" si des matériels différents devaient être employés, notamment une imprimante à largeur de papier réduite.

Dès l'apparition du message "d'accueil", l'opérateur peut utiliser le clavier de l'ORIC comme celui d'une machine à écrire. La touche RETURN permet de déclencher un "retour chariot" (à la ligne), mais l'opération s'effectue automatiquement si une tentative est faite d'entrer plus de 66 caractères à la suite. Un signal sonore est alors émis. Il est conseillé d'éviter de laisser agir cette sécurité, car une coupure de mot intervient alors de façon arbitraire. Si une erreur est commise, la touche DEL permet d'effacer tout l'écran, et de reprendre la dernière ligne (ligne en cours) en entier. Le reste du texte demeure bien sûr en mémoire même s'il disparaît de l'écran.

L'action sur la touche ESCAPE, à n'importe quel moment, est interprétée comme la fin de la saisie du texte.

Tout le contenu de la mémoire est alors transféré sur l'imprimante, sous forme de lignes de 66 caractères au maximum. Après l'édition de chaque exemplaire, la machine propose une nouvelle copie. Une réponse de type oui/non a été prévue à chaque fois, mais une simple boucle FOR/NEXT pourrait permettre l'enchaînement automatique d'un nombre donné d'impressions.

Il faudrait alors ajuster le nombre de lignes imprimables à la longueur exacte des feuilles de papier utilisées, car il existe diverses exécutions du classique "paravent".

• Conclusion

Bien que ce programme, utilisé tel qu'il est fourni ici, puisse déjà rendre des services, il doit plutôt être considéré comme une "pièce détachée" destinée à s'insérer dans toutes sortes de logiciels personnels de traitement de texte sur imprimante.

Son principal intérêt est de fournir un moyen commode d'échapper à la restriction imposée par l'ORIC en matière de longueur de chaînes de caractères.

Etendu jusqu'aux limites de la capacité mémoire de la machine, ce procédé utilisant les tableaux de chaînes pourrait permettre la mémorisation d'un nombre respectable de "pages" dactylographiées.

7056 valeurs de résistances dans 84 tiroirs

La série E 12 comporte, entre un ohm et 8,2 mégohms, 84 valeurs normalisées, couramment disponibles dans n'importe quel laboratoire d'électronique avec une tolérance souvent bien meilleure que 5 %.

Il est fréquent d'éprouver le besoin impératif d'une valeur non standard, qui pourra cependant être obtenue par une association série ou parallèle d'éléments courants.

Pour des raisons pratiques, l'association en parallèle s'avère plus commode, malgré une plus grande complexité des calculs devant être exécutés.

Même en se limitant à l'association de deux résistances, on obtient directement 7056 valeurs comprises entre 0,5 ohm et 4,1 mégohms.

Comme il ne saurait être question de connaître cette série par cœur (!), nous avons mis au point ce petit logiciel capable d'effectuer très rapidement une recherche qui ne laissera aucune possibilité au hasard.

Principe du programme

```
20 CLS:PRINT:PRINT
30 PRINT "VALEUR A OBTENIR (EN OHMS) ?"
32 INPUT R
34 PRINT "TOLERANCE ADMISSIBLE EN % ?"
36 INPUT T
50 CLS:PRINT:PRINT
60 PRINT "RECHERCHE EN COURS ..."
70 DIM A(84) : C=1
80 E=1 : GOSUB 1000
90 E=10 : GOSUB 1000
100 E=100 : GOSUB 1000
110 E=1000 : GOSUB 1000
120 E=10000 : GOSUB 1000
130 E=100000 : GOSUB 1000
140 E=1000000 : GOSUB 1000
145 IF R<.5 THEN PRINT "IMPOSSIBLE" :
EXPLODE : STOP
```

```
150 B=1
160 IF A(B)>R THEN 200
170 B=B+1
180 IF B>84 THEN PRINT "IMPOSSIBLE" :
EXPLODE : STOP
190 GOTO 160
200 FOR G=B TO 84
210 FOR H= B TO 84
220 X=(A(G)*A(H))/(A(G)+A(H))
230 IF X>R+(R*T/100) THEN 300
240 IF X<R-(R*T/100) THEN 300
250 PRINTA(G);" OHMS" : PING
260 PRINT "EN PARALLELE AVEC"
270 PRINTA(H);" OHMS"
280 PRINT:PRINT "EQUIVALENT A:"
290 PRINT(INT(100*X))/100;"OHMS"
295 PRINT:PRINT"POUR CONTINUER:
PRESSER UNE TOUCHE" : GET Z$
300 NEXT H
310 NEXT G
320 PRINT "PAS TROUVE ..." : EXPLODE : STOP
330 REM COPYRGHT 1983 P.GUEULLE
1000 FOR F=1 TO 12
1010 READ D
1020 A(C)=D*E
1030 C=C+1
1040 NEXT
1050 RESTORE
1060 RETURN
1070 DATA 1,1.2,1.5,1.8,2.2,2.7,3.3,3.9,4.7,
5.6,6.8,8.2
```

La méthode que nous avons programmée en machine consiste à calculer tous les assemblages susceptibles de présenter une résistance voisine de celle demandée, et à retenir le premier se situant dans la fourchette de tolérance spécifiée par l'utilisateur. Les éventuelles autres combinaisons possibles peuvent être consultées à la suite, en cas de manque dans les valeurs de base nécessaires.

Sachant que la résistance d'un montage parallèle est toujours inférieure à la plus faible de ses composantes, la recherche ne com-

mence que par la mise en parallèle des deux valeurs immédiatement supérieures à celle désirée, ce qui accélère très sensiblement le processus.

Sur un ordinateur ORIC, il faut en effet cinq minutes et demi pour une recherche complète, alors que dans les cas raisonnablement simples, une première solution est proposée en quelques secondes seulement. Bien sûr, le délai de réponse augmente avec la précision demandée, mais n'oublions pas que chercher à obtenir une précision de 1 % en associant des résistances à 5 % relève du charlatanisme pur et simple ! En toute rigueur, il faudrait lancer la recherche en spécifiant une tolérance égale ou supérieure à celle des éléments de base qui seront employés. Cela n'est cependant pas une règle absolue, puisque les techniques modernes de fabrication des résistances à couche permettent souvent d'excellentes surprises lors du contrôle de composants réputés à 5 %.

Pour des raisons de simplification, c'est la machine qui calcule les 84 valeurs de base à partir des douze coefficients normalisés, contenus dans une ligne DATA.

Ces 84 valeurs prennent place dans un tableau de variables DIM A (84), ce qui permet un accès rapide et facile à chacune d'elles lors des calculs, grâce à une double boucle FOR-NEXT.

Mis au point sur un ordinateur ORIC, ce programme devrait fonctionner sans modifications sensibles sur toutes les machines munies d'un BASIC de type MICROSOFT ou similaire. Seules les instructions sonores PING et EXPLODE pourront éventuellement poser un problème de transcription. Ces signalisations ne sont d'ailleurs nullement indispensables, mais bien pratiques pour alerter l'opérateur lors de recherches de durée non négligeable.

PING indique qu'un résultat est affiché, alors qu'EXPLODE signale l'échec des recherches, soit à cause d'une valeur demandée hors intervalle, soit du fait d'exigences trop strictes en matière de tolérance.

Des découvertes intéressantes pourront être faites en prenant le "risque" d'exiger une valeur exacte (tolérance 0 %) : les téléphonistes seront ainsi ravis de constater que 1000 et 1500 ohms en parallèle donnent exactement 600 ohms...

Utilisation du logiciel

Le mode d'emploi du programme est des plus simples, et conviendra même aux utilisateurs non informaticiens.

Sitôt lancé par RUN, le logiciel demande la valeur devant être obtenue, et la tolérance admissible. Les deux réponses seront bien sûr validées par RETURN.

Un message s'affiche alors, destiné à faire patienter l'utilisateur qui n'aura plus qu'à attendre un signal sonore. Si une solution apparaît à l'écran, elle répond obligatoirement aux conditions imposées. Une simple pression sur la barre d'espace suffit pour que la recherche continue, et que d'éventuelles autres solutions soient proposées, toujours dans la même fourchette de tolérance.

Pour arrêter le programme en cours de déroulement, il faut lancer un BREAK grâce à la combinaison CTRL C. Le poussoir RESET dissimulé sous la machine peut aussi être utilisé, puisqu'il n'efface pas le programme.

On peut cependant tout simplement emmener le programme jusqu'à son terme, auquel cas il s'arrête seul sur un STOP. Dans tous les cas, un RUN permettra de relancer l'exécution pour procéder à une nouvelle recherche.

Conclusion

Malgré sa simplicité, ce programme est un exemple d'utilisation pratique "rentable" d'un ordinateur individuel. La machine ne prend que cinq minutes et demi pour effectuer une recherche complète qui occuperait pendant *une trentaine d'heures* un opérateur muni d'une calculette à quatre opérations !

Le traitement automatisé ne laisse aucune place à l'improvisation, et ne néglige aucune possibilité : si une solution existe, elle sera immanquablement trouvée.

Ce principe de recherche systématique des possibilités est le même que celui servant dans les jeux d'échecs, et permettant à l'ordinateur de gagner presque à tout coup. Seulement, il ne s'agit pas là d'un jeu...

Un télétexte personnel sur ORIC !

Les systèmes de télétexte permettent de visualiser sur l'écran d'un terminal relié à un réseau de communication, des pages d'informations provenant d'une banque de données. L'ORIC peut servir de base à la mise sur pied d'un petit système expérimental utilisant un réseau téléphonique ou un canal radio "CB", ouvrant ainsi les portes de l'univers "télématique" à l'utilisateur de petites machines individuelles.

Côté pile et côté face du télétexte

Un système de télétexte comprend, de part et d'autre du réseau de communication, des équipements radicalement différents. Côté utilisateur, un terminal aussi simple et économique que possible (par exemple le MINITEL des PTT, en location), éventuellement complété par des périphériques tels qu'écran couleur ou imprimante de recopie d'écran.

Il ne serait théoriquement pas impossible de transformer un système ORIC en terminal TELETEL, mais l'ampleur considérable des transcodages à prévoir oblige à y réfléchir à deux fois ! Côté banque de données, on peut aussi bien trouver un centre informatique occupant un immeuble entier qu'un micro-ordinateur de table posé sur le coin d'un bureau !

Il existe des "centres serveurs" fonctionnant fort bien sur un simple ordinateur "Goupil".

Cela étant posé, on peut songer à construire un système utilisant la même architecture, mais dans lequel centre serveur et terminaux distants seraient bâtis autour d'ORICS, et communiqueraient selon une procédure nullement normalisée, mais accessible à l'amateur.

Il ne faudra évidemment pas attendre d'un tel système expérimental des performances dignes de TELETEL, bien que des applications pratiques intéressantes puissent être trouvées sans peine !

Applications d'un télétexte d'amateur

Par définition, un réseau de télétexte sert à mettre les ressources d'une base de données unique à la disposition d'un certain nombre d'utilisateurs géographiquement dispersés. C'est donc en priorité aux clubs, associations, et autres groupements que s'adressera notre propos.

On peut facilement imaginer de remplacer un bulletin périodique imprimé à grands frais, par une suite de pages vidéo accessibles à travers un répondeur téléphonique ou diffusées dans certains créneaux horaires sur une fréquence radio bien précise (par exemple un canal CB ou même une mini-radio libre). L'avantage du système (propre à la notion de télétexte), est qu'une mise à jour de l'information diffusée peut être faite à tout moment et en très peu de temps (une seule page à recomposer), sans attendre la prochaine parution d'un bulletin mensuel ou, pire encore, trimestriel !

Il est très facile d'automatiser entièrement la diffusion, ce qui évite d'avoir à monopoliser une personnne à cet effet.

Le système proposé

Après divers essais, nous avons renoncé à la création d'un système véritablement *interactif* qui, comme TELETEL, devrait permettre un *dialogue* entre terminal et base de données. Nous avons préféré un système inspiré d'ANTIOPE, dans lequel *toutes* les pages existantes sont diffusées en permanence les unes derrière les autres, la sélection étant assurée par le terminal lui-même. Bien sûr le temps d'accès à une page précise en pâtit quelque peu, mais il est possible de diffuser plus souvent les pages les plus demandées (par exemple la page "menu" pourra être intercalée toutes les deux ou trois pages spécifiques). De toute façon, ce système est le seul qui permette une diffusion par radio à plusieurs utilisateurs de façon simultanée.

Le principe du système consiste à transmettre le signal sonore élaboré par la sortie cassette de l'ORIC lors d'un CSAVE du seul fichier d'affichage (situé entre les adresses décimales 48000 et 49119). La transmission d'un écran entier prend à peu près dix secondes en mode rapide (une minute et demi en mode lent !), ce qui peut être considéré comme très honorable.

Seul point noir, le rechargement d'un écran sur l'ORIC bloque carrément la machine sur une erreur "out of memory". Nous verrons cependant que cette anomalie est tolérable.

Les différentes pages de textes et/ou de graphismes seront enregistrées les unes à la suite des autres sur une cassette en boucle sans fin (TDK ENDLESS), faciles à trouver dans une grande variété de durées chez les revendeurs HIFI.

Cette cassette sera alors, soit placée dans un répondeur téléphonique, soit lue dans l'entrée "micro" d'un émetteur radio fonctionnant sur la fréquence choisie. Dans les deux cas, il importe évidemment de s'assurer que l'exploitation qui est faite du système reste compatible avec les réglementations en vigueur dans le domaine des télécommunications.

Du côté du "terminal", l'utilisation est fort simple, puisqu'il suffit de frapper CLOAD"" pour visualiser la première page qui se présentera.

Ce procédé n'est toutefois guère orthodoxe, puisque l'on charge ainsi un écran comme un vulgaire programme, sans même spécifier à quelle adresse mémoire il doit être implanté ! D'ailleurs, un inesthétique "Ready" vient se superposer à l'image utile.

L'emploi d'un court logiciel de "réception" est donc préférable.

Côté “technique”, on peut bien sûr songer à relier directement l’entrée cassette de l’ORIC à la sortie BF d’un récepteur radio ou d’un amplificateur téléphonique. La qualité de l’interface sonore de l’ORIC s’accommode en effet de signaux même assez maltraités.

Il peut cependant s’avérer plus commode, surtout dans le cas d’une coûteuse liaison téléphonique interurbaine, d’enregistrer sur cassette les informations transmises, pour les recharger en différé. Un simple capteur téléphonique à ventouse fera alors merveille, tant avec un poste PTT qu’à proximité du haut-parleur d’un récepteur FM, CB, ou autre, en l’absence de prise d’enregistrement.

En ce qui concerne la construction de répondeurs téléphoniques adaptés à cet usage, on se reportera à notre ouvrage *Interphone, téléphone, montages périphériques,* paru chez le même éditeur.

Il importe de noter que la qualité sonore “en bout de ligne” n’est pas aussi bonne qu’à la sortie d’un magnétophone, et que d’éventuels parasites risquent de causer des erreurs de transmission. En radio, notamment, on ne comptera pas sur des portées utiles dépassant la moitié de celles pouvant être obtenues en phonie.

Un logiciel de création de pages vidéo

Il ne suffit pas de pouvoir transmettre des pages de télétexte, encore faut-il les créer !

La mise en page harmonieuse de textes et de graphismes simplifiés obtenus par association de caractères ASCII est facile et rapide à l’aide de ce programme :

```
10 CLS : PLOT 2,0,"NOUVELLE PAGE"
20 A$=KEY$
30 IF A$=CHR$(13) THEN 100
40 PRINT A$;
50 GOTO 20
100 PLOT  2,0,"NOM DE CETTE PAGE ?"
105 PRINT CHR$(19); CHR$(17)
110 INPUT N$
115 PRINT CHR$(19); CHR$(17)
120 PLOT 2,0,"
130 PLOT 2,0,"PAGE: " + N$
140 CSAVE N$,A48000,E49119
150 GOTO 10
160 REM COPYRIGHT 1983 P.GUEULLE
```

L'utilisation de ce logiciel est très simple : dès le lancement par RUN, l'indication "nouvelle page" apparaît en haut d'un écran vide.

Le curseur peut être déplacé grâce aux touches fléchées, et les caractères frappés seront inscrits à la droite de ce curseur. La touche DEL peut servir, comme à l'accoutumée, à corriger toute erreur.

La touche RETURN *ne commande pas* le retour à la ligne, qui est automatique, mais déclenche la sauvegarde sur cassette, à la fin de laquelle la création d'une nouvelle page est proposée.

Avant l'enregistrement proprement dit, la machine réclame le nom de la page : c'est sous ce même nom qu'elle devra être relue par la suite.

Il est d'usage de commencer par une page nommée "MENU", qui fournira la liste de toutes les pages disponibles avec leur nom exact.

A condition de bien prendre ses repères, on pourra venir "écraser" une page périmée, par un écran remis à jour. C'est là un avantage décisif du télétexte, que de permettre une actualisation aussi fréquente que nécessaire des données offertes à la consultation.

Un logiciel pour le "terminal"

Pour le confort de l'utilisateur, il est commode de charger le logiciel suivant en machine, avant de chercher à consulter une page vidéo :

```
10 CLS
20 PLOT 3,0,"NOM DE LA PAGE ?"
30 PRINT CHR$(10); CHR$(10)
40 INPUT N$
50 PRINT CHR$(19); CHR$(17)
60 CLOAD N$, A48000,E49119
70 REM COPYRIGHT 1983 P. GUEULLE
```

La page désirée sera tout simplement appelée par son nom *exact* (on s'abstiendra d'orthographier en minuscules un nom initialement composé en majuscules), et l'écran correspondant s'affichera dès qu'il sera trouvé. Pendant tout le délai d'attente nécessaire, le message habituel "searching" persistera en haut de l'écran.

L'interface cassette de l'ORIC est suffisamment fiable pour ne pas se formaliser si des signaux non conformes lui parviennent entretemps : parole, musique, bruits divers, ou même programmes pris en cours de route : un bon point pour de telles applications ! Un

petit blâme, par contre, pour le blocage de la machine après un chargement d'écran : il faut en effet ré-entrer le logiciel de "réception" chaque fois qu'une nouvelle page est à consulter...

Un écran de toutes les couleurs

Certaines publicités relatives à l'ORIC font état de seize couleurs, peut-être parce que huit couleurs d'encre peuvent se combiner à huit couleurs de papier, mais nous devons avouer ne jamais être arrivé à atteindre ce chiffre de seize en comptant les nuances disponibles, qui sont *noir, rouge, vert, jaune, bleu, magenta, cyan, et blanc...* Par contre, un petit artifice fort simple permet d'obtenir à peu près toutes les teintes imaginables, de l'orange au marron en passant par le beige et le mauve !

Après tout, n'importe quel téléviseur normalement constitué peut reconstituer une foule de couleurs à partir de ses trois canons rouge, vert et bleu...

• Les mystères du code ASCII 126

Dans le manuel de l'ORIC, la liste des caractères ASCII s'arrête au code 125.

La curiosité étant souvent récompensée, lançons un :

PRINT CHR$ (126)

et constatons qu'un petit carré gris apparaît.

En noir et blanc "informatique", le gris est habituellement obtenu en faisant voisiner de très petits points noirs et blancs, comme sur les photographies de presse. Et précisément, le caractère 126 ressemble, de près, à un petit damier.

Les cases blanches sont bien sûr de la couleur du "papier", alors que les cases noires épousent la couleur de "l'encre". Et si nous altérions ces deux couleurs ?

En télévision couleur, le jaune est obtenu en juxtaposant un point rouge et un point vert. Avec l'ORIC, nous pouvons juxtaposer huit couleurs d'encre et huit couleurs de papier. Certes ces deux jeux de teintes sont identiques, mais comme les cases noires du damier sont un peu plus grandes que les cases blanches (à cause du temps de réponse des circuits vidéo du récepteur), nous n'obtiendrons pas exactement la même nuance si nous échangeons tout simplement les couleurs d'encre et de papier...

Sur un total théorique de 8×8 soit 64 nuances possibles, il faut retrancher huit combinaisons "doubles" pour lesquelles les couleurs d'encre et de papier sont les mêmes (en fait, les couleurs de base de l'ORIC).

Il reste tout de même 56 possibilités distinctes !

Bien sûr, certaines des teintes obtenues ne sont pas très agréables, bien sûr certaines produisent une sorte de scintillement sur l'écran, mais il reste un très vaste domaine à explorer !

• Un logiciel multicolore

```
10 FOR F=80 TO 87
20 FOR G=64 TO 71
30 PRINT CHR$(27); CHR$(F);
40 PRINT CHR$(27); CHR$(G);
50 PRINT CHR$(27); CHR$(72);
60 FOR H=1 TO 36
70 PRINT CHR$(126);
80 NEXT H
90 PRINT
100 NEXT G
110 NEXT F
120 REM COPYRIGHT 1983 P.GUEULLE
```

Ce petit programme fait défiler sur l'écran TV des bandes de toutes les couleurs pouvant être obtenues par la méthode décrite.

Il utilise les classiques (et rébarbatifs !) attributs série ainsi que l'inévitable code ESCAPE (CHR$ (27).

Cependant, il est bien rare d'avoir besoin d'autant de couleurs à la fois.

A la limite, il suffit souvent de spécifier une couleur INK, une couleur PAPER, et de savoir que des carrés d'une troisième couleur pourront être placés n'importe où sur l'écran grâce à des PLOT X, Y, CHR$ (126).

L'avantage du procédé est qu'aucun attribut n'est mis à contribution, avec toutes les lourdeurs que cela suppose. Il est tout de même important de noter que la même combinaison peut mener à des résultats très différents d'un récepteur à l'autre, en fonction des réglages, de la marque du tube cathodique, et de l'âge du poste. Des différences n'apparaissant guère sur les couleurs "primaires" deviennent évidentes sur certains mélanges.

Qui n'a jamais constaté, dans un magasin à grande surface, l'ampleur des différences affectant par exemple les teints des visages d'un poste à un autre, alors que la classique mire est reproduite avec une imperturbable qualité ?

Quoi qu'il en soit, ce petit artifice peut améliorer très nettement la présentation de bien des images composées par l'ORIC !

Chapitre 9

Pour aller plus loin

Tout au long de notre tour d'horizon des possibilités de l'ORIC, nous avons découvert progressivement les instructions les plus importantes de son BASIC.

Les logiciels que nous avons fournis en tant qu'exemples sont là pour prouver que des applications de toutes sortes peuvent être développées à partir de ces bases.

Nos lecteurs ayant fait l'effort de suivre attentivement l'itinéraire qui leur a été proposé dans cet ouvrage doivent désormais être capables d'élaborer leurs programmes personnels, en vue de satisfaire leurs besoins les plus divers.

Cependant, il est possible d'aller encore beaucoup plus loin dans l'exploitation des possibilités de cet ordinateur.

Utilisation de logiciels du commerce

Avec le développement explosif de l'informatique individuelle se sont créées de nombreuses officines de commercialisation de logiciels, tant en France qu'en Grande-Bretagne, où le phénomène a pris naissance.

Ces sociétés servent en fait à mettre en contact les créateurs de logiciels, souvent isolés, et les utilisateurs potentiels, qui ne le sont parfois guère moins.

Mais surtout, on trouve dans le commerce des programmes représentant une somme considérable de travail, au point d'être irréalisables par une personne seule.

Un logiciel de "vingt années-homme" représente, par exemple, le travail de quarante personnes pendant six mois, ou... d'un unique programmeur pendant vingt ans !

Il ne faut donc pas se leurrer : à la vitesse où évoluent ces techniques, à la vitesse à laquelle se périment les bonnes idées, les logiciels complexes ne peuvent qu'être des œuvres collectives.

L'utilisateur ne peut envisager d'écrire lui-même que des programmes relativement simples, faute de quoi ses besoins risqueraient d'avoir évolué entre-temps !

On trouve pour l'ORIC une quantité sans cesse croissante de logiciels répondant à un large éventail de besoins, depuis les inévitables jeux d'arcade jusqu'à des systèmes complets de gestion de fichiers.

La plupart de ces programmes font appel à des instructions évoluées du BASIC ou même au langage machine, si ce n'est à des langages spéciaux...

Jusqu'au bout du BASIC

Le BASIC dont est équipé l'ORIC est une version étendue du classique BASIC MICROSOFT, que l'on rencontre sur bien d'autres machines. Les principales différences se situent dans le domaine des graphismes et du son.

Le manuel d'origine de l'ORIC ne traite pas en détail toutes les fonctions disponibles, et des habitués d'autres ordinateurs travaillant sous MICROSOFT arrivent à découvrir sur l'ORIC des fonctions dont il n'est même pas fait mention dans la documentation.

Dans cet ouvrage d'initiation, nous avons également dû opérer des choix pour ne pas écraser le lecteur.

L'ORIC dispose d'une panoplie assez complète de fonctions mathématiques, dont l'emploi ne pose pas plus de problèmes que sur une calculatrice scientifique : trigonométrie, logarithmes, exponentielles, valeur absolue, signe, racine carrée, et à condition de bien la chercher, la puissance. Une très commode possibilité de définition de fonctions (DEF FN) permet à l'utilisateur d'expliquer une fois pour toutes à la machine les modalités d'un calcul.

Un nom est alors donné à cette formule, qui pourra être réutilisée à loisir, sous ce même nom : PRINT FNA (X), par exemple.

Des effets graphiques plus évolués peuvent être créés à l'aide des ordres FILL et POKE, capables d'aller agir directement dans la mémoire-écran de l'ordinateur. La lourdeur du procédé lui fait perdre beaucoup de son intérêt, même s'il permet, à l'extrême rigueur, d'obtenir plusieurs couleurs en haute résolution.

En fait, ces possibilités doivent être mises à contribution lorsque, pour la résolution d'un problème précis, les instructions du BASIC courant ne suffisent plus.

Le langage machine

Il est des tâches que le BASIC, même utilisé jusqu'au bout de ses ressources, n'arrivera jamais à accomplir.

D'une façon générale, toutes les actions extrêmement rapides ou contraires à "l'esprit" du BASIC devront être exécutées à l'aide d'un autre langage.

Le LANGAGE MACHINE est le seul dialecte que comprend le microprocesseur 6502 équipant l'ORIC. Ce composant complexe est le véritable "cerveau" de l'ordinateur, puisque c'est lui qui exécute, sur les zéros et les uns, toutes les opérations élémentaires qui, combinées entre elles, constituent le jeu d'instructions du BASIC.

Deux boîtiers de mémoire inaltérable (ROM) contiennent, dans l'ORIC, un programme entièrement écrit en lagage machine, dont l'exécution commence dès la mise sous tension de l'ordinateur.

C'est lui, et lui seul, qui donne à l'ORIC le comportement que nous lui connaissons. Si nous découvrons des anomalies dans le fonctionnement de la machine, c'est à des erreurs dans l'écriture de ce

programme qu'il faut les attribuer. L'une des tâches de ce long programme est de "traduire" en langage machine les instructions BASIC introduites par l'opérateur. En mode commande, cette traduction ou "interprétation", est exécutée dès que la touche RETURN est pressée. Si l'instruction n'est pas conforme à la *syntaxe* du BASIC, c'est le programme "interpréteur" qui se charge d'en avertir l'opérateur.

En mode programme, les instructions sont traduites au fur et à mesure de leur exécution : si, dans une boucle par exemple, une instruction est exécutée mille fois, elle sera traduite mille fois ! Là se situe le principal frein à la rapidité d'exécution d'un programme BASIC.

Egalement, le BASIC MICROSOFT, aussi performant soit-il, ne représente qu'un nombre limité de combinaisons d'instructions machine. On pourrait en imaginer d'autres, et c'est précisément ce qu'il y a lieu de faire lorsque l'on atteint les limites de la programmation BASIC.

L'ORIC dispose de plusieurs instructions permettant au programmeur d'ordonner à la machine d'interrompre l'exécution de l'"interpréteur BASIC". Le microprocesseur 6502 part alors exécuter un programme en langage machine qui est implanté, dans la mémoire de la machine, à partir d'une adresse que l'opérateur a bien voulu lui indiquer.

Il peut tout simplement s'agir d'une "routine" contenue dans la "ROM", mais il n'y a là guère d'innovation. Un programmeur maîtrisant bien le langage machine du 6502 peut écrire un programme machine de toutes pièces, et le loger dans un espace de la mémoire "RAM" qu'il se sera réservé au moyen de l'instruction HIMEM.

Ce programme se composera d'une liste d'OCTETS, c'est-à-dire de nombres décimaux compris entre 0 et 255, ou de nombres hexadécimaux allant de 0 à FF, ou encore de nombres binaires compris entre 00000000 et 11111111.

La fonction POKE du BASIC permet d'écrire en RAM un octet donné à une adresse donnée. DOKE en écrit deux à la fois, ce qui peut gagner du temps.

PEEK permet de "lire" une adresse, et DEEK, d'en inspecter deux à la fois.

Les octets peuvent être introduits dans l'ORIC sous forme décimale, ou hexadécimale en les faisant précéder du signe #. Les habitudes de chacun pourront dont être conciliées !

Une fois le programme machine entré en mémoire, les instructions CALL et USR permettent de faire quitter à la machine le BASIC au profit de ce programme, et de procéder à des échanges de données entre les deux langages.

C'est le langage machine qui, lors de l'exécution d'une instruction spécialement prévue par le programmeur, rendra de nouveau les commandes à l'interpréteur BASIC. Faute de cette instruction, la machine resterait bloquée jusqu'à un providentiel RESET manuel !

Un programme écrit en langage machine fonctionne en complète indépendance vis-à-vis du BASIC. *Tout est permis* à un programmeur habile qui pourrait, à l'extrême limite, écrire un nouvel interpréteur BASIC.

En effet, ne le cachons pas, la programmation en langage machine est fort complexe, en tout cas, bien plus que le BASIC. Son apprentissage suppose la lecture de nombreux ouvrages qui ne comptent pas parmi les plus simples. D'ailleurs, le 6502 n'a pas la réputation du microprocesseur le plus agréable à programmer...

De plus, il est difficile d'accéder aux informations sur l'organisation interne de l'ORIC, pourtant vitales pour la programmation rationnelle de son microprocesseur. Comme toujours en informatique, il faut éviter de brûler les étapes : la programmation de l'ORIC en langage machine est possible et pleine de promesses, mais il faut l'aborder de façon très méthodique, et ne pas sous-estimer la somme de travail à fournir, pas plus que le temps à y consacrer !

Vers d'autres langages évolués

Nous avons vu que le BASIC présente certaines limitations, mais que le langage machine est pratiquement tout-puissant. En revanche, l'apprentissage du BASIC est facile, alors que beaucoup d'efforts sont exigés du candidat à la programmation machine.

Une solution intermédiaire existe, qui offre une très grande souplesse et qui ouvre d'immenses horizons à l'ORIC, comme d'ailleurs à n'importe quel ordinateur individuel.

Imaginons un grand programme machine, chargé, mettons à partir d'une cassette (mais qui pourrait aussi l'être par enfichage d'une "cartouche"), et qui obligerait l'ORIC à travailler dans un langage tout différent : PASCAL, LOGO, ou FORTH.

Eh bien de telles cassettes existent, du moins pour le FORTH, et

existeront, n'en doutons pas, pour d'autres langages dans les mois à venir. Peut-être même existeront-elles déjà lorsque ce livre sortira de presse, tout va si vite ! L'adaptation, généralement peu coûteuse, d'un nouveau langage à un ordinateur est une complète métamorphose : nouveau manuel, nouveau comportement du clavier, de l'écran, nouveaux mots et nouvelle syntaxe à apprendre. Bref, une toute nouvelle machine à découvrir.

FORTH est aussi différent que possible du BASIC : son apprentissage n'est guère plus difficile, mais la "conversion" est parfois délicate : les modes de raisonnement utilisés sont souvent diamétralement opposés.

Par contre, la machine devient capable d'exécuter comme en se jouant, des tâches sur lesquelles le BASIC aurait achoppé. Bref, encore de passionnantes découvertes en perspective... en attendant les suivantes !

Achevé d'imprimer sur les presses de l'Imprimerie Marcel Bon 70001 Vesoul
Dépôt légal : mars 1984 - N° d'éditeur : 417 - N° d'imprimeur : 2763

PILOTEZ VOTRE ORIC :
ORIC 1 et ORIC ATMOS

L'ORIC 1 a su s'imposer comme le symbole de l'ordinateur individuel de la « seconde génération ».
L'ORIC ATMOS dispose de toutes les fonctions (notamment graphiques et sonores) de son prédécesseur, dont les divers « défauts de jeunesse » ont été habilement corrigés.
Cet ouvrage s'adresse aussi bien aux **débutants** de l'informatique, qu'aux **habitués d'autres machines,** désireux de se convertir à l'ORIC ou à l'ATMOS.
Loin de se limiter à une simple initiation, Patrick Gueulle va jusqu'à traiter des plus récents **circuits d'interfaces** permettant de transformer l'ORIC ou l'ATMOS en téléphone à annuaire incorporé ou en oscilloscope à mémoire.

Principaux chapitres :

- Prise de contact avec l'ORIC
- Le magnétophone
- Diverses applications
- Instructions sonores et raccordements
- Couleurs et haute résolution
- Imprimante et traitement de texte
- L'ORIC robot : les interfaces

ISBN 2-85535-067-0

TARIF CODE L

10 PILOTEZ VOTRE ORIC P. GUEULLE ETSF